Allen-Bradley

Controladores Logix5000

(Num. cat.) 1756-L1, -L1Mx

Manual de referencia del conjunto de instrucciones generales

Información importante para el usuario

Debido a la variedad de usos de los productos descritos en esta publicación, las personas responsables de la aplicación y uso de este equipo de control deben asegurarse de que se hayan seguido todos los pasos necesarios para que cada aplicación y uso cumplan con todos los requisitos de rendimiento y seguridad, incluyendo leyes, normativas, códigos y normas aplicables.

Los ejemplos de ilustraciones, gráficos, programas y esquemas mostrados en esta guía tienen la única intenció de ilustrar el testo. Debido a las muchas variables y requisitos asociados con cualquier instalación particular, Allen-Bradley no puede asumir responsabilidad u obligación (incluyendo responsabilidad de propiede intelectual) por el uso real basado en los ejemplos mostrados en esta publicación.

La publicación de Allen-Bradley, publicación SGI-1.1, *Safety Guidelines for the Application, Installation and Maintenance of Solid-State Control* (disponible en la oficina de Allen-Bradley local), describe algunas diferencias importantes entre equipos transistorizados y dispositivos electromecánicos, las cuales deben tomarse en consideración al usar productos tales como los descritos en esta publicación.

Está prohibida la reproducción total o parcial de los contenidos de esta publicación de propiedad exclusiva sin el permiso escrito de Rockwell Automation.

En este manual hacemos anotaciones para advertirle sobre consideraciones de seguridad:

ATENCION: Identifica información o prácticas o circunstancias que pueden conducir a lesiones personales o la muerte, daños materiales o pérdidas económicas.

Las notas de “Atención” le ayudan a:

- identificar un peligro
- evitar un peligro
- reconocer las consecuencias

Importante: Identifica información crítica para una correcta aplicación y entendimiento del producto.

Identifica un consejo o nota. Sírvase tomar nota de que en esta publicación se usa el punto decimal para separar la parte enter de la decimal de todos los númos.

Allen-Bradley, ControlLogix, DH+, Logix5000, Logix5550, MicroLogix, PLC-2, PLC-3, PLC-5, Rockwell Software, RSLinx, RSNetWorx y SLC son marcas comerciales de Rockwell Automation.

ControlNet es una marca comercial de ControlNet International, Ltd.

DeviceNet es una marca comercial de Open DeviceNet Vendor Association.

Ethernet es una marca comercial de Digital Equipment Corporation, Intel y Xerox Corporation.

Introducción

Esta versión del documento contiene nueva información actualizada. Además, se ha eliminado alguna información o se incluye en otro manual.

Información actualizada

La siguiente tabla indica los cambios más significativos efectuados en este documento desde la versión más reciente:

Para obtener esta información nueva o actualizada:	**Vea el capítulo:**
Cómo especificar los detalles de comunicación (ficha Comunicación) de una instrucción de mensaje	3
Cómo obtener acceso al objeto TASK	3
Cómo formatear expresiones	4, 5 y 7
El uso de los operadores ABS, MOD y TRN	4, 5 y 7
Fracciones en el resultado de un DIV y SQR	5
El valor absoluto y las instrucciones de módulo	5
La instrucción de truncar	15
Valores inmediatos	Apéndice A
Conversiones de datos	Apéndice A
Los tiempos de ejecución de instrucciones	Apéndice C
Requisitos de memoria de extensión con signo	Apéndice D
Requisitos de memoria de relleno con ceros	Apéndice D

Información eliminada

La siguiente tabla indica la información que ha sido eliminada de este manual pero que se puede encontrar en otros manuales:

Para obtener información acerca de:	**Vea este manual:**
Instrucciones de movimiento	*Logix5000 Controllers Motion Instruction Set Reference Manual,* publicación 1756-6.4.3
Terminología Logix5000	*Manual del usuario de controladores Logix5000*, publicación 1756-6.5.12ES

Notas:

Dónde se encuentran las instrucciones

Use la tabla siguiente para encontrar las instrucciones que aparecen en este manual. Si ve las letras ***MIM*** al lado de una instrucción, vea el documento *Logix5000 Controllers Motion Instruction Set Reference Manual*, publicación 1756-6.4.3, para obtener más información acerca de la instrucción.

Instrucción:	Página o manual:
ABS	5-19
ACS	13-10
ADD	5-6
AFI	10-15
AND	6-9
ASN	13-8
ATN	13-12
AVE	7-35
BRK	11-5
BSL	8-2
BSR	8-5
BTD	6-5
BTR (tipo MSG)	3-2
BTW (tipo MSG)	3-2
CLR	6-8
CMP	4-2
COP	7-28
COS	13-4
CPT	5-2
CTD	2-14
CTU	2-11
DDT	12-9
DEG	15-2
DIV	5-12
DTR	12-16
EQU	4-6
FAL	7-7
FBC	12-2
FFL	8-8
FFU	8-14
FLL	7-32
For	11-2
FRD	15-6
FSC	7-18

Instrucción:	Página o manual:
GEQ	4-8
GRT	4-10
GSV	3-27
JMP	10-2
JSR	10-4
LBL	10-2
LEQ	4-12
LES	4-14
LFL	8-20
LFU	8-26
LIM	4-16
LN	14-2
LOG	14-4
MAAT	***MIM***
MAFR	***MIM***
MAG	***MIM***
MAH	***MIM***
MAHD	***MIM***
MAJ	***MIM***
MAM	***MIM***
MAPC	***MIM***
MAR	***MIM***
MAS	***MIM***
MASD	***MIM***
MASR	***MIM***
MATC	***MIM***
MAW	***MIM***
MCCP	***MIM***
MCD	***MIM***
MCR	10-11
MDF	***MIM***
MDO	***MIM***
MDR	***MIM***
MDW	***MIM***

Instrucción:	Página o manual:
MEQ	4-19
MGPS	***MIM***
MGS	***MIM***
MGSD	***MIM***
MGSP	***MIM***
MGSR	***MIM***
MOD	5-14
MOV	6-2
MRAT	***MIM***
MRHD	***MIM***
MRP	***MIM***
MSF	***MIM***
MSG	3-2
MSO	***MIM***
MUL	5-10
MVM	6-3
NEG	5-18
NEQ	4-22
NOP	10-16
NOT	6-15
ONS	1-9
OR	6-11
OSF	1-14
OSR	1-11
OTE	1-6
OTL	1-7
OTU	1-8
PID	12-19
RAD	15-3
RES	2-18
RET	10-4, 11-6
RTO	2-8
SBR	10-4
SIN	13-2

Instrucción:	Página o manual:
SQI	9-2
SQL	9-11
SQO	9-6
SQR	5-16
SRT	7-39
SSV	3-27
STD	7-42
SUB	5-8
TAN	13-6
TND	10-10
TOD	15-4
TOF	2-5
TON	2-2
TRUN	15-8
UID	10-13
UIE	10-14
XIC	1-2
XIO	1-4
XOR	6-13
XPY	14-6

Cómo usar este manual

Introducción

Este manual forma parte de varios documentos acerca de ControlLogix.

Tarea/meta:	Documentos:
Instalación del controlador y los componentes del mismo	*Inicio rápido del controlador Logix5550*, publicación 1756-10.1ES *Instrucciones de instalación de la tarjeta de memoria Logix5550*, publicación 1756-5.33ES
Uso del controlador	*Logix5000 Controllers User Manual*, publication 1756-6.5.12
Programar el controlador para aplicaciones secuenciales **Usted está aquí**	*Manual de referencia del conjunto de instrucciones generales Logix5000*, publicación 1756-6.4.1ES
Programar el controlador para aplicaciones de movimiento	*Logix5000 Controllers Motion Instruction Set Reference Manual,* publication 1756-6.4.3
Configuración y comunicación con los módulos de E/S digitales	*Digital Modules User Manual*, publicación 1756-6.5.8
Configuración de los módulos de E/S analógicas	*Analog Modules User Manual*, publicación 1756-6.5.9
Configuración y uso de los módulos de movimiento	*ControlLogix Motion Module User Manual*, publicación 1756-6.5.16
Selección e instalación de un chasis	*ControlLogix Chassis Installation Instructions*, publicación 1756-5.69
Selección e instalación de una fuente de alimentación eléctrica	*Instrucciones de instalación de la fuente de alimentación eléctrica ControlLogix, publicación 1756-5.1ES*
Importar un archivo o tags de texto en un proyecto	*Logix5550 Controller Import/Export Reference Manual, publication 1756-6.8.4*
Exportar un proyecto o tags a un archivo de texto	
Convertir una aplicación PLC-5 ó SLC 500 a una aplicación Logix5000	*Logix5550 Controller Converting PLC-5 or SLC 500 Logic to Logix5550 Logic Reference Manual, publication 1756-6.8.5*

Quién debe usar este manual

Este documento proporciona al programador los detalles acerca de cada instrucción disponible para un controlador Logix5550. Usted ya debe estar familiarizado con la manera en que el controlador Logix5550 almacena y procesa los datos.

Los usuarios sin experiencia deben leer todos los detalles acerca de una instrucción antes de usar ésta. Los programadores con experiencia pueden consultar la información de instrucciones para verificar los detalles.

Propósito de este manual

Este manual proporciona información acerca de cada instrucción que es compatible con los controladores Logix5000. Cada descripción se presenta según el formato siguiente:

Esta sección:	Proporciona este tipo de información:
Nombre de la instrucción	Identifica la instrucción define si la instrucción es para la entrada o la salida
Operandos	indica todos los operandos de la instrucción
Estructura de control	indica los bits y valores de estado de control de la instrucción, si los hay
Descripción	describe el uso de la instrucción define las diferencias entre la instrucción habilitada e inhabilitada, si fuese necesario
Ejecución	define las especificaciones acerca de cómo funciona la instrucción durante: • preescán • condición de entrada de renglón es falsa • condición de entrada de renglón es verdadera
Indicadores de estado aritmético	define si la instrucción afecta o no los indicadores de estado aritmético vea el apéndice A
Condiciones de fallo	define si la instrucción genera o no fallos menores o mayores, de ser sí, define el tipo y código de fallo
Ejemplo	proporciona por lo menos un ejemplo de programación incluye una descripción que explica cada ejemplo

Información común para todas las instrucciones

El conjunto de instrucciones Logix5000 tiene algunos atributos comunes:

Esta información:	Vea este apéndice:
atributos comunes	el apéndice A define: • indicadores de estado aritmético • tipos de datos • palabras clave
arreglos	el apéndice B define los arreglos y explica cómo el controlador manipula los mismos
estructuras	el apéndice C ilustra las estructuras de control compatibles con el controlador

Convenciones y términos afines

Establecer y restablecer

Este manual usa los términos establecer y restablecer para definir el estado de los bits (booleanos) y valores (no booleanos):

Este término:	Significa:
establecer	el bit está establecido en 1 (ON) el valor está establecido en cualquier número que no sea cero
restablecer	el bit está restablecido en 0 (OFF) todos los bits en un valor están restablecidos en 0

En la sección de operandos, los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Condición del renglón

El controlador evalúa las instrucciones de lógica de escalera según la condición de renglón que precede la instrucción (condición de entrada de renglón). Según la condición de de entrada de renglón y la instrucción, el controlador establece la condición de renglón que sigue la instrucción (condición de salida de renglón), lo cual, a su vez, afecta cualquier instrucción subsiguiente.

Si la condición de entrada de renglón de una instrucción de entrada es verdadera, el controlador evalúa la instrucción y establece la condición de renglón de entrada según los resultados de la instrucción. Si la instrucción evaluada es verdadera, la condición de renglón de entrada es verdadera; si la instrucción evaluada es falsa, la condición de renglón de salida es falsa.

Notas:

Instrucciones de bit (XIC, XIO, OTE, OTL, OTU, ONS, OSR, OSF)

Capitulo 1

Introducción 1-1
Examine If Closed (XIC) 1-2
Examine If Open (XIO) 1-4
Output Energize (OTE) 1-6
Output Latch (OTL) 1-7
Output Unlatch (OTU) 1-8
One Shot (ONS) 1-9
One Shot Rising (OSR) 1-11
One Shot Falling (OSF) 1-14

Instrucciones de temporizador y contador (TON, TOF, RTO, CTU, CTD, RES)

Capitulo 2

Introducción 2-1
Timer On Delay (TON) 2-2
Timer Off Delay (TOF) 2-5
Retentive Timer On (RTO) 2-8
Count Up (CTU) 2-11
Count Down (CTD) 2-14
Reset (RES) 2-18

Instrucciones de entrada/ salida (MSG, GSV, SSV)

Chapter 3

Introducción 3-1
Message (MSG) 3-2
MSG Error Codes. 3-7
- Códigos de error ControlLogix (CIP) 3-7
- Códigos de error extendidos ControlLogix 3-8
- Códigos de error PLC y SLC (.ERR). 3-9
- Códigos de error extendidos PLC y SLC (.EXERR). 3-10
- Códigos de error de transferencia en bloques 3-11
- Códigos de error Logix5550 3-12
- Códigos de error extendidos Logix5550 3-12

Cómo especificar los detalles de configuración (ficha Configuración). 3-13
- Cómo especificar los mensajes CIP 3-14
- Cómo usar mensajes CIP genérico para restablecer los módulos de E/S. 3-15
- Cómo especificar los mensajes PLC-5. 3-16
- Cómo especificar los mensajes SLC 3-17
- Cómo especificar los mensajes de transferencia en bloques . . 3-17
- Cómo especificar los mensajes PLC-3. 3-18
- Cómo especificar los mensajes PLC-2. 3-19

Ejemplos de configuración MSG. 3-20
Cómo especificar los detalles de comunicación (ficha Comunicación). 3-21
- Cómo especificar una ruta de conexión 3-21
- Cómo especificar un método de comunicación: 3-25
- Cómo seleccionar una opción de caché: 3-26

Get System Value (GSV) y Set System Value (SSV) 3-27

Objetos GSV/SSV 3-29
Cómo obtener acceso al objeto AXIS 3-30
Cómo obtener acceso al objeto CONTROLLER. 3-37
Cómo obtener acceso al objeto CONTROLLERDEVICE . . 3-37
Cómo obtener acceso al objeto CST 3-39
Cómo obtener acceso al objeto DF1 3-40
Cómo obtener acceso al objeto FAULTLOG 3-43
Cómo obtener acceso al objeto MESSAGE. 3-43
Cómo obtener acceso al objeto MODULE 3-45
Cómo obtener acceso al objeto MOTIONGROUP 3-46
Cómo obtener acceso al objeto PROGRAM 3-47
Cómo obtener acceso al objeto ROUTINE 3-47
Cómo obtener acceso al objeto SERIALPORT. 3-48
Cómo obtener acceso al objeto TASK. 3-49
Cómo obtener acceso al objeto WALLCLOCKTIME 3-50
Ejemplo de programación GSV/SSV. 3-51
Cómo obtener información de fallo 3-51
Cómo establecer los indicadores
de habilitación e inhabilitación 3-52

Chapter 4

Instrucciones de comparación (CMP, EQU, GEQ, GRT, LEQ, LES, LIM, MEQ, NEQ)

Introducción 4-1
Compare (CMP). 4-2
Operadores válidos. 4-4
Cómo formatear expresiones 4-4
Cómo determinar el orden de operación 4-5
Equal to (EQU) 4-6
Greater Than or Equal to (GEQ) 4-8
Greater Than (GRT). 4-10
Less Than or Equal to (LEQ). 4-12
Less Than (LES) 4-14
Limit (LIM) 4-16
Mask Equal to (MEQ) 4-19
Cómo introducir un valor de máscara inmediato. 4-19
Not Equal to (NEQ) 4-22

Chapter 5

Instrucciones de cálculo/ matemáticas (CPT, ADD, SUB, MUL, DIV, MOD, SQR, NEG, ABS)

Introducción 5-1
Compute (CPT) 5-2
Operadores válidos. 5-4
Cómo formatear expresiones 5-4
Cómo determinar el orden de operación 5-5
Add (ADD) 5-6
Subtract (SUB). 5-8
Multiply (MUL). 5-10
Divide (DIV) 5-12
Modulo (MOD) 5-14
Square Root (SQR) 5-16
Negate (NEG) 5-18
Absolute Value (ABS). 5-19

Chapter 6

Instrucciones de transferencia/lógica (MOV, MVM, BTD, CLR, AND, OR, XOR, NOT)

Introducción 6-1
Move (MOV) 6-2
Masked Move (MVM) 6-3
Cómo introducir un valor de máscara inmediato 6-3
Bit Field Distribute (BTD) 6-5
Clear (CLR) 6-8
Bitwise AND (AND) 6-9
Bitwise OR (OR) 6-11
Bitwise Exclusive OR (XOR) 6-13
Bitwise NOT (NOT) 6-15

Chapter 7

Instrucciones de arreglo (archivo)/misceláneos (FAL, FSC, COP, FLL, AVE, SRT, STD)

Introducción 7-1
Cómo seleccionar el modo de operación 7-1
Modo todos 7-2
Modo numérico 7-3
Modo incremental 7-4
File Arithmetic and Logic (FAL) 7-6
Operadores válidos 7-15
Cómo formatear expresiones 7-15
Cómo determinar el orden de operación 7-16
File Search and Compare (FSC) 7-17
Operadores válidos 7-25
Cómo formatear expresiones 7-25
Cómo determinar el orden de operación 7-26
File Copy (COP) 7-27
File Fill (FLL) 7-31
File Average (AVE) 7-34
File Sort (SRT) 7-38
File Standard Deviation (STD) 7-41

Chapter 8

Instrucciones de arreglo (archivo)/desplazamiento (BSL, BSR, FFL, FFU, LFL, LFU)

Introducción 8-1
Bit Shift Left (BSL) 8-2
Bit Shift Right (BSR) 8-5
FIFO Load (FFL) 8-8
FIFO Unload (FFU) 8-14
LIFO Load (LFL) 8-20
LIFO Unload (LFU) 8-26

Chapter 9

Instrucciones de secuenciador (SQI, SQO, SQL)

Introducción 9-1
Sequencer Input (SQI) 9-2
Cómo introducir un valor de máscara inmediato 9-3
Cómo usar SQI sin SQO 9-5
Sequencer Output (SQO) 9-6
Cómo introducir un valor de máscara inmediato 9-7
Cómo usar SQI con SQO 9-9
Cómo restablecer la posición de SQO 9-10
Sequencer Load (SQL) 9-11

Chapter 10

Instrucciones de control de programa (JMP, LBL, JSR, RET, SBR, TND, MCR, UID, UIE, AFI, NOP)

Introducción 10-1
Jump to Label (JMP)
Label (LBL) 10-2
Jump to Subroutine (JSR)
Subroutine (SBR)
Return (RET) 10-4
Temporary End (TND) 10-10
Master Control Reset (MCR) 10-11
User Interrupt Disable (UID) 10-13
User Interrupt Enable (UIE) 10-14
Always False (AFI) 10-15
No Operation (NOP) 10-16

Chapter 11

Instrucciones FOR/interrupción (FOR, BRK, RET)

Introducción 11-1
For (FOR) 11-2
Break (BRK) 11-5
Return (RET) 11-6

Chapter 12

Instrucciones especiales (FBC, DDT, DTR, PID)

Introducción 12-1
File Bit Comparison (FBC) 12-2
Cómo seleccionar el modo buscar 12-4
Diagnostic Detect (DDT) 12-9
Cómo seleccionar el modo buscar 12-11
Data Transitional (DTR) 12-16
Cómo introducir un valor de máscara inmediato 12-16
Proportional Integral Derivative (PID) 12-19
Cómo configurar una instrucción PID 12-24
Cómo especificar el ajuste 12-24
Cómo especificar la configuración 12-25
Cómo especificar alarmas 12-25
Cómo especificar la escala 12-26
Cómo usar las instrucciones PID 12-26
Bloqueo de acción integral y transferencia sin perturbaciones de manual a automático 12-28
Temporización de la instrucción PID 12-29

Reinicio sin perturbaciones 12-32
Uniformidad de derivada 12-34
Cómo establecer la banda muerta 12-34
Cómo usar el límite de salida 12-35
Prealimentación o polarización de salida 12-35
Lazos en cascada 12-35
Cómo controlar una relación 12-36

Chapter 13

Instrucciones trigonométricas (SIN, COS, TAN, ASN, ACS, ATN)

Introducción 13-1
Sine (SIN) 13-2
Cosine (COS) 13-4
Tangent (TAN) 13-6
Arc Sine (ASN) 13-8
Arc Cosine (ACS) 13-10
Arc Tangent (ATN) 13-12

Chapter 14

Instrucciones matemáticas avanzadas (LN, LOG, XPY)

Introducción 14-1
Natural Log (LN) 14-2
Log Base 10 (LOG) 14-4
X to the Power of Y (XPY) 14-6

Chapter 15

Instrucciones de conversión matemática (DEG, RAD, TOD, FRD, TRN)

Introducción 15-1
Degrees (DEG) 15-2
Radians (RAD) 15-3
Convert to BCD (TOD) 15-4
Convert to Integer (FRD) 15-6
Truncate (TRN) 15-8

Appendix A

Atributos comunes

Introducción A-1
Palabras clave de estado aritmético A-1
Si el tipo de datos es SINT A-2
Si el tipo de datos es INT A-2
Si el tipo de datos es DINT A-2
Otras palabras clave A-4
Tipos de datos A-5
Valores inmediatos A-6
Conversiones de datos A-6
SINT o INT a DINT A-7
Número entero a REAL A-9
DINT a SINT o INT A-9
REAL a un número entero A-9

Appendix B

Conceptos de arreglo

Cómo ver un arreglo como colección de elementos B-1
Cómo indexar a través de los arreglos B-3
Cómo especificar bit dentro de arreglos B-4
Cómo ver un arreglo como un bloque de memoria B-4
Cómo el controlador almacena los datos del arreglo B-5
Cómo variar una dimensión B-6
Asignación de memoria para los arreglos B-6

Appendix C

Tiempo de ejecución

Introducción C-1
Tablas de referencia C-3

Appendix D

Uso de memoria

Introducción D-1
Conversiones de datos D-2
Requisitos de memoria de extensión con signo D-2
Requisitos de memoria de relleno con ceros D-2
Instrucciones D-3
Indices de arreglos D-7

Instrucciones de bit
(XIC, XIO, OTE, OTL, OTU, ONS, OSR, OSF)

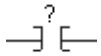
Introducción

Use las instrucciones de bit (tipo relé) para monitorear y controlar el estado de los bits.

Si usted desea:	Use esta instrucción:	Vea la página:
habilitar las salidas cuando se establece un bit	XIC	1-2
habilitar las salidas cuando se restablece un bit	XIO	1-4
establecer un bit	OTE	1-6
establecer un bit (retentivo)	OTL	1-7
restablecer un bit (retentivo)	OTU	1-8
habilitar las salidas para un escán cada vez que un renglón se hace verdadero	ONS	1-9
establecer un bit para un escán cada vez que un renglón se hace verdadero	OSR	1-11
establecer un bit para un escán cada vez que el renglón se hace falso	OSF	1-14

Examine If Closed (XIC)

La instrucción XIC es una instrucción de entrada.



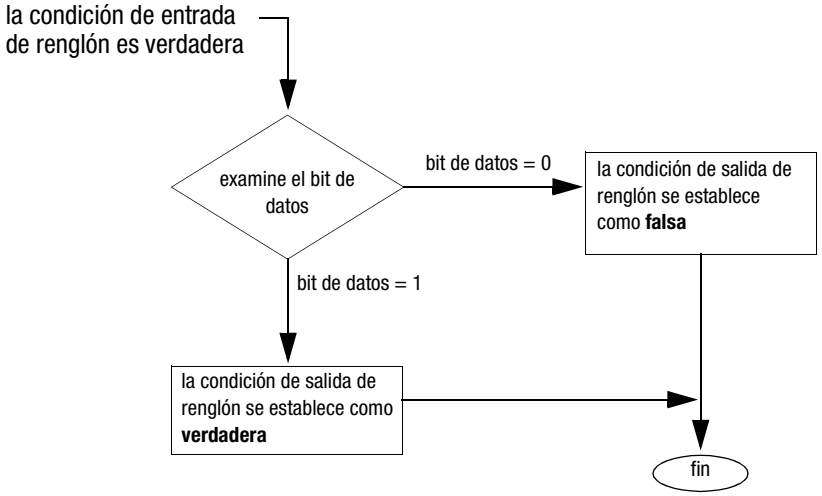
Operandos:

Operando:	Tipo:	Formato:	Descripción:
bit de datos	BOOL	tag	bit que se prueba

Descripción: La instrucción XIC examina el bit de datos para determinar si está establecido.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.

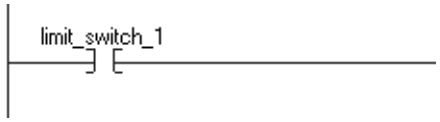

la condición de entrada de renglón es verdadera
examine el bit de datos
bit de datos = 0
la condición de salida de renglón se establece como **falsa**
bit de datos = 1
la condición de salida de renglón se establece como **verdadera**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

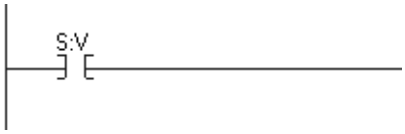
Ejemplo de XIC:

ejemplo 1



Si *limit_switch_1* está establecido, esto habilita la próxima instrucción (la condición de salida de renglón es verdadera).

ejemplo 2



Si *S:V* está establecido (indica que ha ocurrido un overflow), esto habilita la próxima instrucción (la condición de sakuda de renglón es verdadera).

Otros formatos:

Formato:	Sintaxis:
texto neutro	`XIC(data_bit);`
texto ASCII	`XIC data_bit`

Instrucciones relacionadas:

XIO

Examine If Open (XIO)

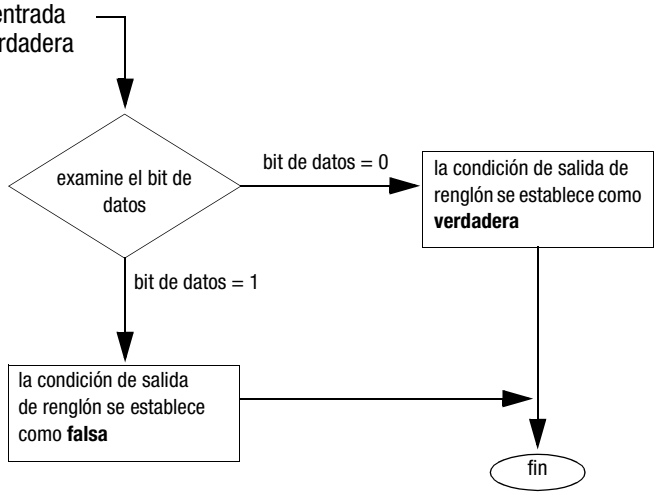
La instrucción XIO es una instrucción de salida.

Operandos:

Operando:	Tipo:	Formato:	Descripción:
bit de datos	BOOL	tag	bit que se prueba

Descripción: La instrucción XIC examina el bit de datos para determinar si está restablecido.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

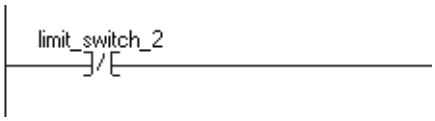
la condición de entrada de renglón es verdadera
examine el bit de datos
bit de datos = 0
la condición de salida de renglón se establece como **verdadera**
bit de datos = 1
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

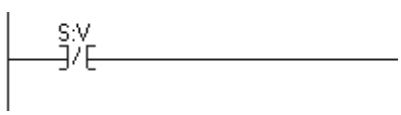
Ejemplo de XIO:

ejemplo 1



Si *limit_switch_2* está restablecido, esto habilita la próxima instrucción (la condición de salida de renglón es verdadera).

ejemplo 2



Si *S:V* está restablecido (indica que no ha ocurrido un overflow), esto habilita la próxima instrucción (la condición de salida de renglón es verdadera).

Otros formatos:

Formato:	Sintaxis:
texto neutro	`XIO(data_bit);`
texto ASCII	`XIO data_bit`

Instrucciones relacionadas:

XIC

Output Energize (OTE)

La instrucción OTE es una instrucción de salida.

Operandos:

Operando:	Tipo:	Formato:	Descripción:
bit de datos	BOOL	tag	bit que se establece o se restablece

Descripción: La instrucción OTE establece o restablece el bit de datos.

Cuando la instrucción OTE está habilitada, el controlador establece el bit de datos. Cuando la instrucción OTE está inhabilitada, el controlador restablece el bit de datos.

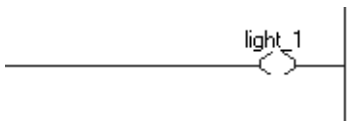
Ejecución:

Condición:	Acción:
preescán	El bit de datos se restablece. La condición de salida de renglón ese establece como falsa.
la condición de entrada de renglón es falsa	El bit de datos se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El bit de datos se establece. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de OTE:



Una vez habilitada, la instrucción OTE establece (enciende) *light_1*.
Una vez inhabilitada, la instrucción OTE restablece (apaga) *light_1*.

Otros formatos:

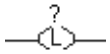
Formato:	Sintaxis:
texto neutro	`OTE(data_bit);`
texto ASCII	`OTE data_bit`

Instrucciones relacionadas: OTL, OTU

Output Latch (OTL)

La instrucción OTL es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
bit de datos	BOOL	tag	bit que se establece

Descripción: La instrucción OTL establece (enclava) el bit de datos.

Una vez habilitada, la instrucción OTL establece el bit de datos. El bit de datos permanece establecido hasta que se restablece, típicamente por una instrucción OTU. Una vez inhabilitada, la instrucción OTL no cambia el estado del bit de datos.

Ejecución:

Condición:	Acción:
preescán	El bit de datos no se modifica. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit de datos no se modifica. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El bit de datos se establece. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de OTL:

Una vez habilitada, la instrucción OTL establece *light_2*. Este bit permanece establecido hasta que se restablece, típicamente por una instrucción OTU.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`OTL(data_bit);`
texto ASCII	`OTL data_bit`

Instrucciones relacionadas: OTU, OTE

Output Unlatch (OTU)

La instrucción OTU es una instrucción de salida.

Operandos:

Operando:	Tipo:	Formato:	Descripción:
bit de datos	BOOL	tag	bit que se restablece

Descripción: La instrucción OTU restablece (desenclava) el bit de datos.

Una vez habilitada, la instrucción OTU restablece el bit de datos. Una vez inhabilitada, la instrucción OTU no cambia el estado del bit de datos.

Ejecución:

Condición:	Acción:
preescán	El bit de datos no se modifica. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit de datos no se modifica. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El bit de datos se restablece. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de OTU:

Una vez habilitada, la instrucción OTU restablece *light_2*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`OTU(data_bit);`
texto ASCII	`OTU data_bit`

Instrucciones relacionadas: OTL, OTE

One Shot (ONS)

La instrucción ONS es una instrucción de entrada.

Operandos:

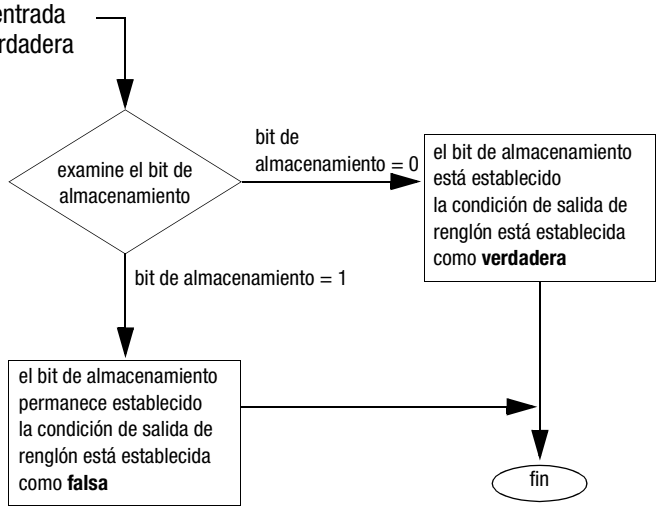
```
   ?
—[ONS]—
```

Operando:	Tipo:	Formato:	Descripción:
bit de almacenamiento	BOOL	tag	bit de almacenamiento interno almacena la condición de renglón de entrada a partir de la última ejecución de la instrucción

Descripción: La instrucción ONS habilita o inhabilita el resto del renglón según el estado del bit de almacenamiento.

Una vez habilitada y cuando se restablece el bit de almacenamiento, la instrucción ONS habilita el resto del renglón. Una vez inhabilitada y o cuando se establece el bit de almacenamiento, la instrucción ONS inhabilita el resto del renglón.

Ejecución:

Condición:	Acción:
preescán	El bit de almacenamiento se establece para evitar un disparo no válido durante el primer escán. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit de almacenamiento se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de ONS: Típicamente una instrucción de entrada precede la instrucción ONS puesto que la instrucción ONS se escanea cuando está habilitada así como inhabilitada para que funcione correctamente. Una vez que la instrucción ONS está habilitada, la condición de entrada de renglón debe hacerse falsa o el bit de almacenamiento se debe restablecer para que la instrucción ONS vuelva a habilitarse.

Este renglón no afecta cualquier escán para el cual se restablece *limit_switch_1* o se establece *storage_1*. En cualquier escán para el cual se establece *limit_switch_1* y se restablece *storage_1*, la instrucción ONS establece *storage_1* y la instrucción ADD incrementa *sum* por 1. Siempre que *limit_switch_1* permanezca establecido, *sum* sigue siendo el mismo valor. El *limit_switch_1* se debe restablecer y volver a establecerse para que *sum* se incremente nuevamente.

Otros formatos:

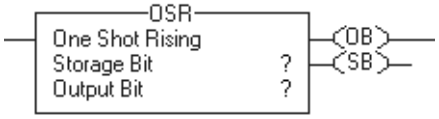
Formato:	Sintaxis:
texto neutro	`ONS(storage_bit);`
texto ASCII	`ONS storage_bit`

Instrucciones relacionadas: OSR, OSF

One Shot Rising (OSR)

La instrucción OSR es una instrucción de salida.

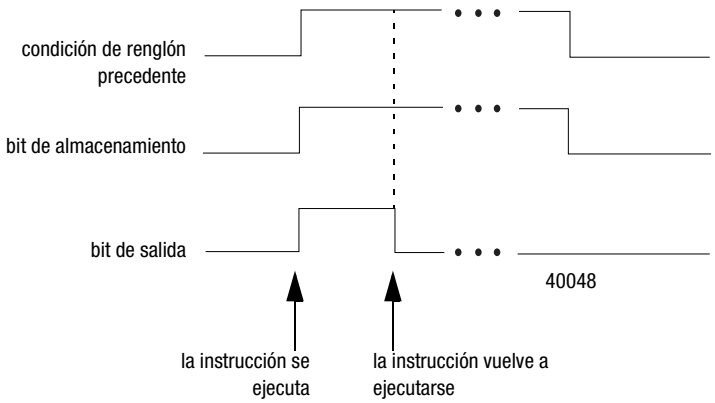
Operandos:



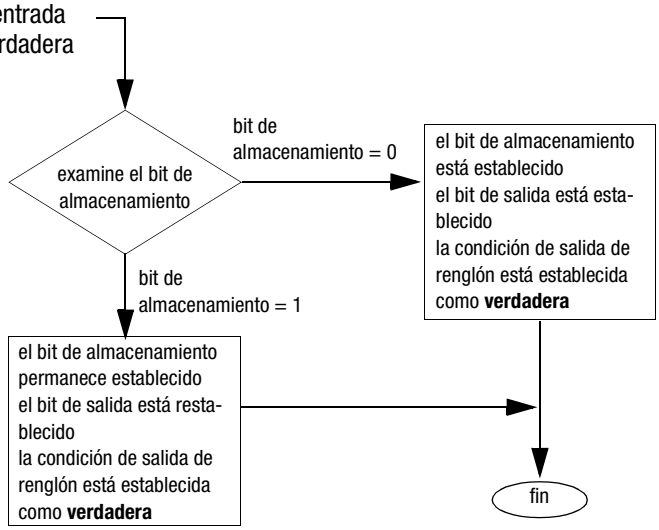
Operando:	Tipo:	Formato:	Descripción:
bit de almacenamiento	BOOL	tag	bit de almacenamiento interno almacena la condición de entrada de renglón a partir de la última ejecución de la instrucción
bit de salida	BOOL	tag	bit que se establece

Descripción: La instrucción OSR establece o restablece el bit de salida según el estado del bit de almacenamiento.

Una vez habilitada y cuando se restablece el bit de almacenamiento, la instrucción OSR establece el bit de salida. Una vez habilitada y el bit de almacenamiento está establecido o una vez inhabilitada, la instrucción OSR restablece el bit de salida.



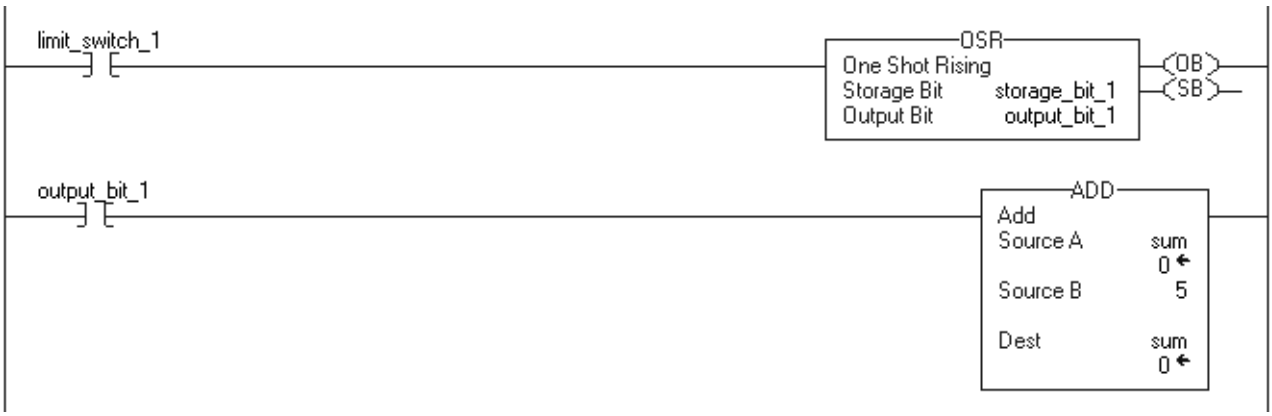
Ejecución:

Condición:	Acción:
preescán	El bit de almacenamiento se establece para evitar un disparo no válido durante el primer escán. El bit de salida se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit de almacenamiento se restablece. El bit de salida no se modifica. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de OSR:



Cada vez que *limit_switch_1* va de restablecido a establecido, la instrucción OSR establece *output_bit_1* y la instrucción ADD incrementa *sum* por 5. Siempre que *limit_switch_1* permanezca establecido, *sum* sigue siendo el mismo valor. El *limit_switch_1* se debe restablecer y volver a establecerse para que *sum* se incremente nuevamente. Se puede usar *output_bit_1* en renglones múltiples para activar otras operaciones.

Otros formatos:

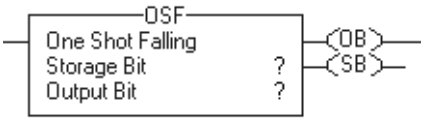
Formato:	Sintaxis:
texto neutro	`OSR(storage_bit,output_bit);`
texto ASCII	`OSR storage_bit output_bit`

Instrucciones relacionadas: OSF, ONS

One Shot Falling (OSF)

La instrucción OSF es una instrucción de salida.

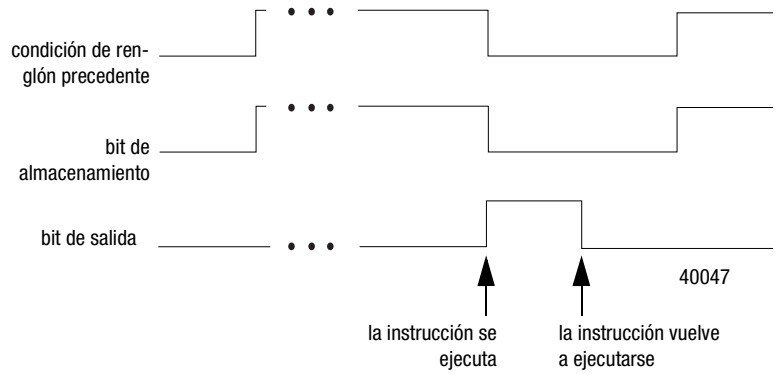
Operandos:



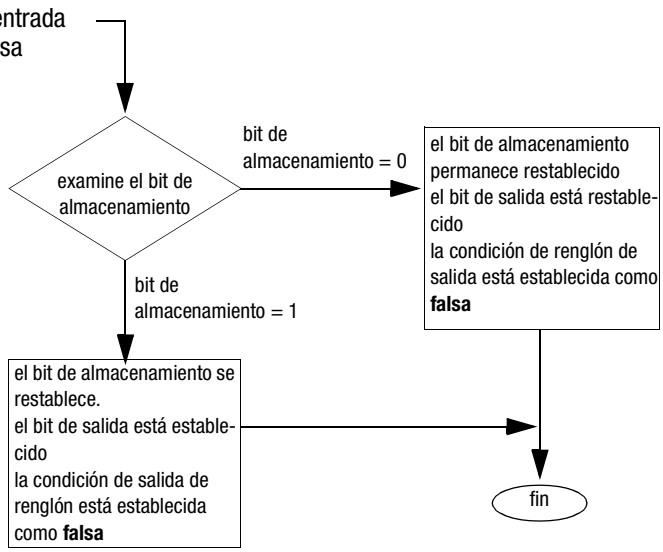
Operando:	Tipo:	Formato:	Descripción:
bit de almacenamiento	BOOL	tag	bit de almacenamiento interno almacena la condición entrada de renglón a partir de la última ejecución de la instrucción
bit de salida	BOOL	tag	bit que se establece

Descripción: La instrucción OSF establece o restablece el bit de salida según el estado del bit de almacenamiento.

Una vez inhabilitada y cuando se establece el bit de almacenamiento, la instrucción OSF establece el bit de salida. Una vez inhabilitada y el bit de almacenamiento está restablecido o una vez inhabilitada, la instrucción OSF restablece el bit de salida.



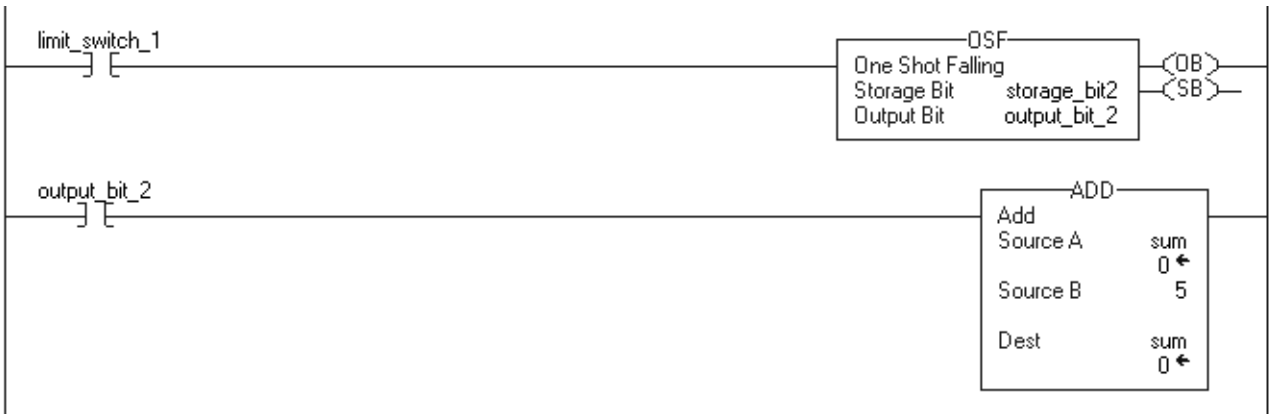
Ejecución:

Condición:	Acción:
preescán	El bit de almacenamiento se restablece para evitar un disparo no válido durante el primer escán. El bit de salida se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	
la condición de entrada de renglón es verdadera	El bit de almacenamiento se establece. El bit de salida se restablece. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de OSF:



Cada vez que *limit_switch_1* va de establecido a restablecido, la instrucción OSR establece *output_bit_2* y la instrucción ADD incrementa *sum* por 5. Siempre que *limit_switch_1* permanezca restablecido, *sum* sigue siendo el mismo valor. El *limit_switch_1* se debe establecer y volver a restablecerse para que *sum* se incremente nuevamente. Se puede usar *output_bit_2* en renglones múltiples para activar otras operaciones.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`OSF(storage_bit,output_bit);`
texto ASCII	`OSF storage_bit output_bit`

Instrucciones relacionadas: OSR, ONS

Instrucciones de temporizador y contador
(TON, TOF, RTO, CTU, CTD, RES)

Introducción

Los temporizadores y contadores controlan las operaciones según el tiempo o el número de eventos.

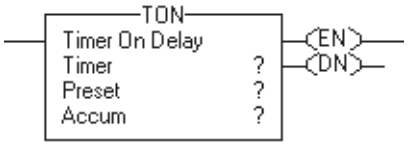
Si usted desea:	Use esta instrucción:	Vea la página:
temporizar la duración de un temporizador habilitado	TON	2-2
temporizar la duración de un temporizador inhabilitado	TOF	2-5
acumular el tiempo	RTO	2-8
contar progresivamente	CTU	2-11
contar regresivamente	CTD	2-14
restablecer un temporizador o contador	RES	2-18

La base de tiempo para todos los temporizadores es 1 mseg.

Timer On Delay (TON)

La instrucción TON es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Temporizador	TIMER	tag	Estructura del temporizador
Preseleccionado	DINT	valor inmediato	la duración del retardo (tiempo acumulado)
Acumulador	DINT	valor inmediato	el total de mseg durante el cual el temporizador ha contado el valor inicial es típicamente 0

Estructura TIMER:

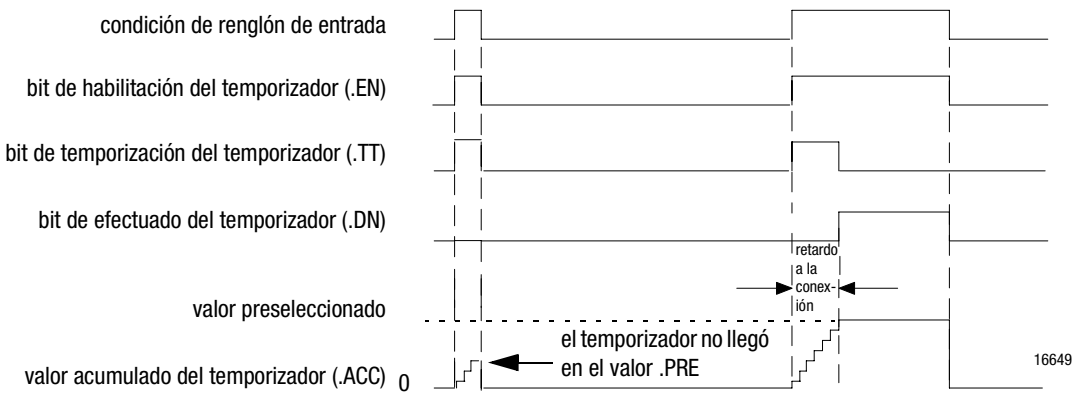
Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción TON está habilitada.
.TT	BOOL	El bit de temporización indica que hay una operación de temporización en progreso.
.DN	BOOL	El bit de efectuado se establece cuando .ACC ≥ .PRE.
.PRE	DINT	El valor preseleccionado especifica el valor (unidades de 1 mseg) que el acumulador debe alcanzar antes de que la instrucción establezca el bit .DN.
.ACC	DINT	El valor acumulado especifica el número de milisegundos que han transcurrido desde la habilitación de la instrucción TON.

Descripción:

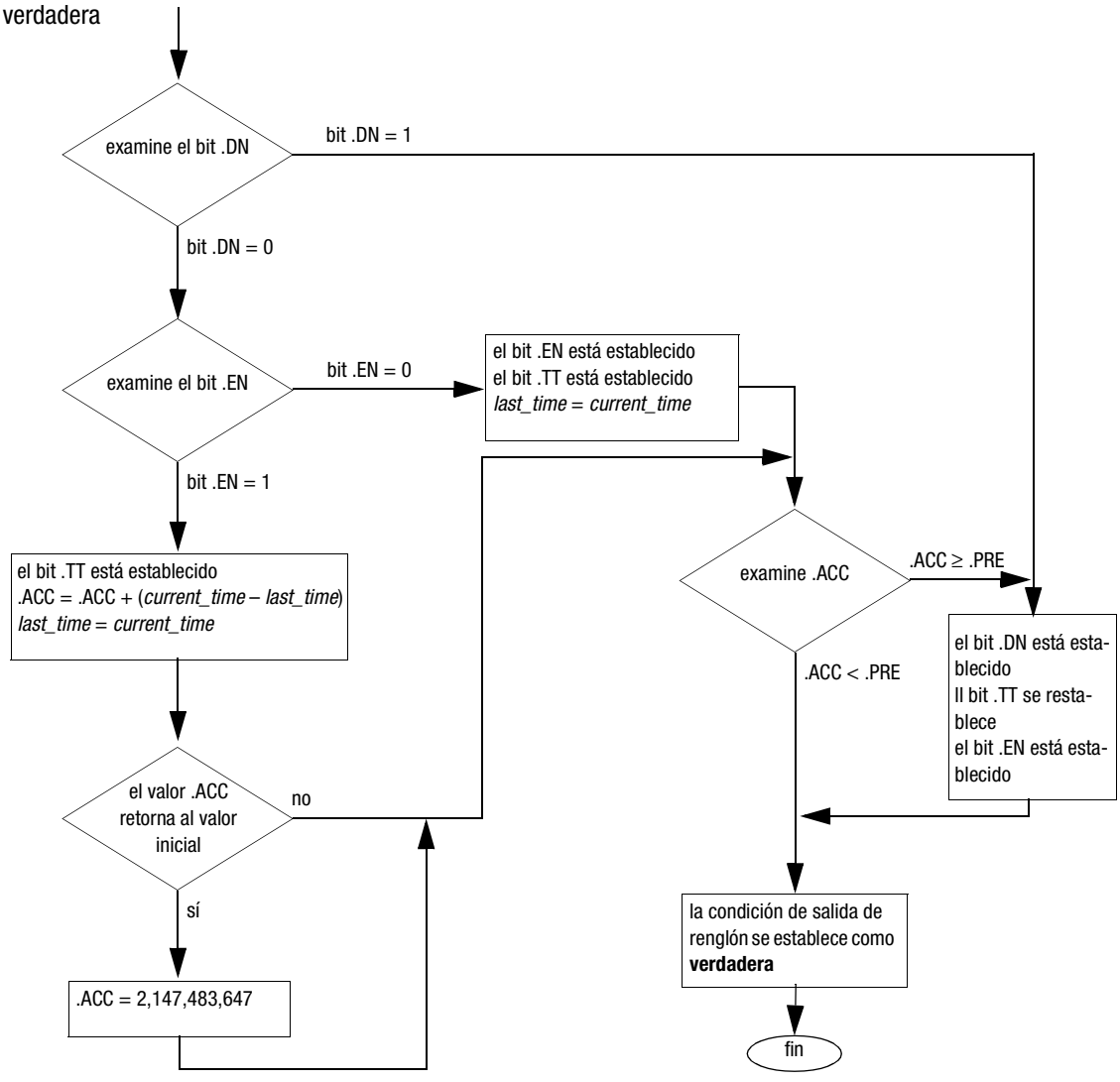
La instrucción TON es un temporizador no retentivo que acumula el tiempo cuando la instrucción está habilitada (la condición de entrada de renglón es verdadera). La base de tiempo siempre es 1 mseg. Por ejemplo, para un temporizador de 2 segundos, introduzca 2000 para el valor .PRE.

Una vez habilitada, la instrucción TON acumula el tiempo hasta que:
- la instrucción TON se inhabilita
- el .ACC ≥ .PRE

Cuando la instrucción TON está inhabilitada, el valor .ACC se restablece.


condición de renglón de entrada
bit de habilitación del temporizador (.EN)
bit de temporización del temporizador (.TT)
bit de efectuado del temporizador (.DN)
valor preseleccionado
valor acumulado del temporizador (.ACC)
0
retardo a la conex-ión
el temporizador no llegó en el valor .PRE
16649

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .TT se restablece. El bit .DN se restablece. El valor .ACC se restablece. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	El bit .EN se restablece. El bit .TT se restablece. El bit .DN se restablece. El valor .ACC se restablece. La condición de salida de renglón se establece como falsa.
la condición de renglón de salida es verdadera	

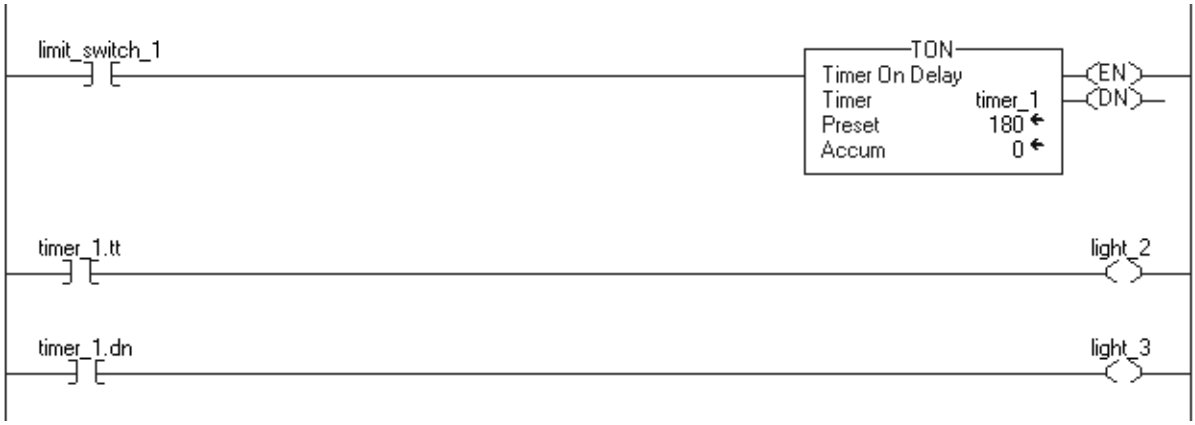
examine el bit .DN
bit .DN = 1
bit .DN = 0
examine el bit .EN
bit .EN = 0
el bit .EN está establecido
el bit .TT está establecido
last_time = current_time
bit .EN = 1
el bit .TT está establecido
.ACC = .ACC + (current_time – last_time)
last_time = current_time
examine .ACC
.ACC ≥ .PRE
el bit .DN está establecido
Il bit .TT se restablece
el bit .EN está establecido
.ACC < .PRE
el valor .ACC retorna al valor inicial
no
sí
.ACC = 2,147,483,647
la condición de salida de renglón se establece como verdadera
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.PRE < 0	4	34
.ACC < 0	4	34

Ejemplo de TON:



Cuando *limit_switch_1* se establece, *light_2* está activado durante 180 mseg (*timer_1* está temporizando). Cuando *timer_1.acc* llega a 180, *light_2* se desactiva y *light_3* se activa. *Light_3* permanece activado hasta que la instrucción TON se inhabilita. Si *limit_switch_1* se restablece mientras *timer_1* temporiza, *light_2* se desactiva.

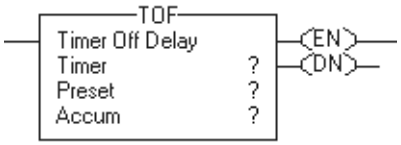
Otros formatos:

Formato:	Sintaxis:
texto neutro	`TON(timer,preset,accum);`
texto ASCII	`TON(timer,preset,accum)`

Instrucciones relacionadas: TOF, RTO

Timer Off Delay (TOF)

La instrucción TOF es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Temporizador	TIMER	tag	Estructura del temporizador
Preseleccion-ado	DINT	valor inmediato	la duración del retardo (tiempo acumulado)
Acumulador	DINT	valor inmediato	el total de mseg durante el cual el temporizador ha contado el valor inicial es típicamente 0

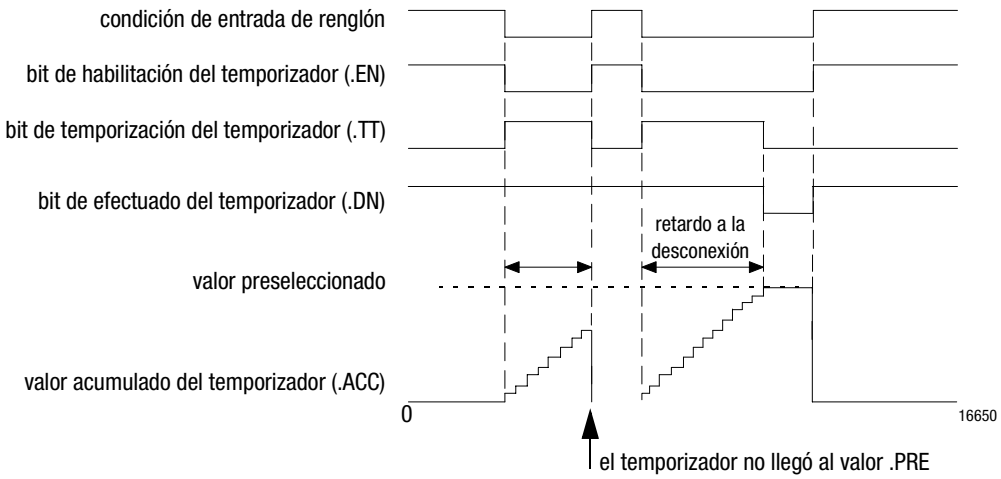
Estructura TIMER:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción TOF está habilitada.
.TT	BOOL	El bit de temporización indica que hay una operación de temporización en progreso.
.DN	BOOL	El bit de efectuado se restablece cuando .ACC ≥ .PRE.
.PRE	DINT	El valor preseleccionado especifica el valor (unidades de 1 mseg) que el acumulador debe alcanzar antes de que la instrucción restablezca el bit .DN.
.ACC	DINT	El valor acumulado especifica el número de milisegundos que han transcurrido desde la habilitación de la instrucción TOF.

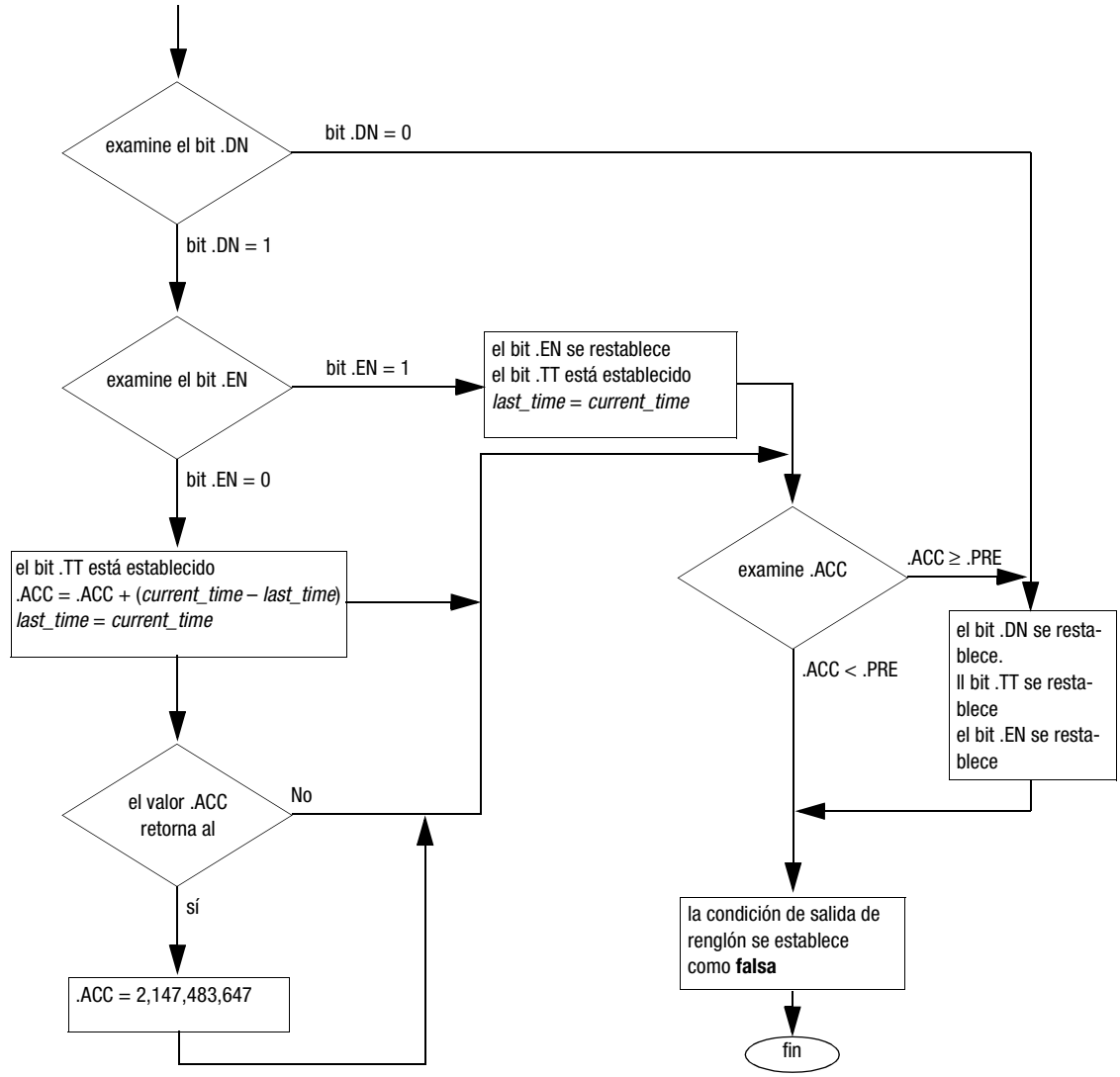
Descripción: La instrucción TOF es un temporizador no retentivo que acumula el tiempo cuando la instrucción está habilitada (la condición de entrada de renglón es falsa). La base de tiempo siempre es 1 mseg. Por ejemplo, para un temporizador de 2 segundos, introduzca 2000 para el valor .PRE.

Una vez habilitada, la instrucción TOF acumula el tiempo hasta que:
- la instrucción TOF se inhabilita
- el .ACC ≥ .PRE

Cuando la instrucción TOF está inhabilitada, el valor .ACC se restablece.


condición de entrada de renglón
bit de habilitación del temporizador (.EN)
bit de temporización del temporizador (.TT)
bit de efectuado del temporizador (.DN)
retardo a la desconexión
valor preseleccionado
valor acumulado del temporizador (.ACC)
0
16650
el temporizador no llegó al valor .PRE

Ejecución:

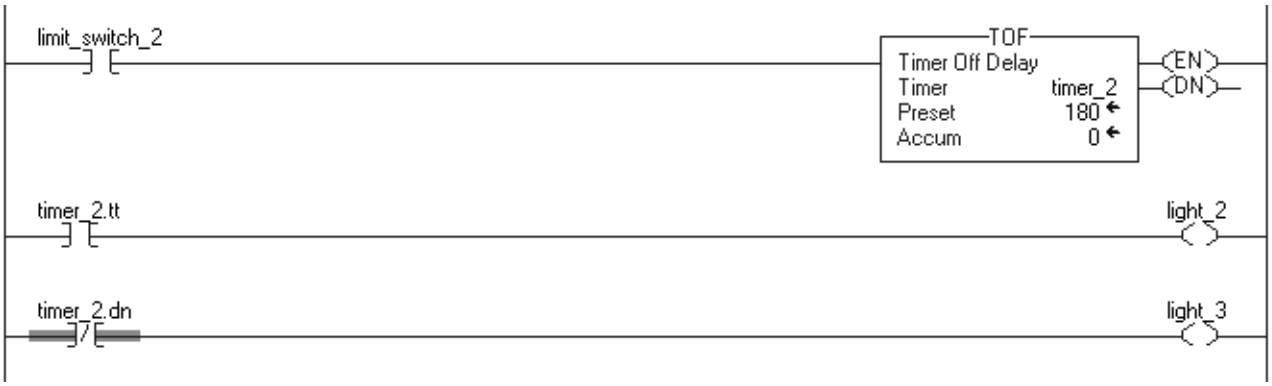
Condición:	Acción:
preescán	El bit .EN se restablece. El bit .TT se restablece. El bit .DN se restablece. El valor .ACC está establecido para ser igual al valor .PRE. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	
la condición de salida de renglón es verdadera	El bit .EN se establece. El bit .TT se restablece. El bit .DN se establece. El valor .ACC se restablece. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.PRE < 0	4	34
.ACC < 0	4	34

Ejemplo de TOF:



Cuando *limit_switch_2* se establece, *light_2* está activado durante 180 mseg (*timer_2* está temporizando). Cuando *timer_2.acc* llega a 180, *light_2* se desactiva y *light_3* se activa. *Light_3* permanece activado hasta que la instrucción TOF se habilita. Si *limit_switch_2* se restablece mientras *timer_2* temporiza, *light_2* se desactiva.

Otros formatos:

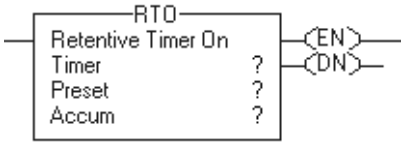
Formato:	Sintaxis:
texto neutro	`TOF(timer,preset,accum);`
texto ASCII	`TOF timer preset accum`

Instrucciones relacionadas: TON, RTO

Retentive Timer On (RTO)

La instrucción RTO es una instrucción de salida.

Operandos:



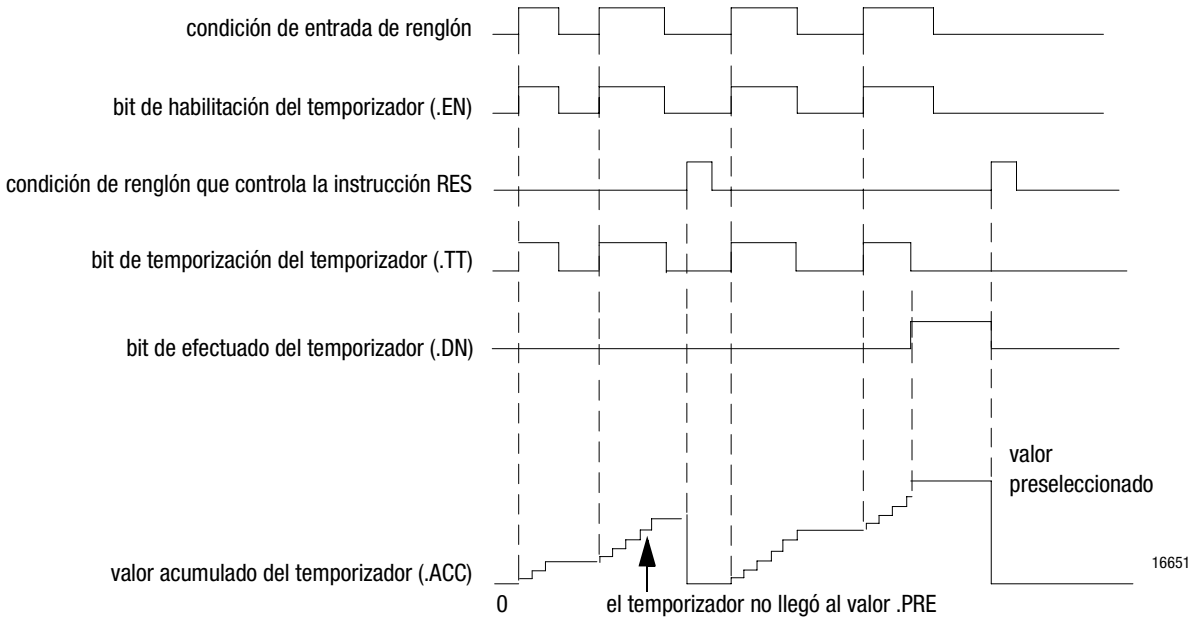
Operando:	Tipo:	Formato:	Descripción:
Temporizador	TIMER	tag	Estructura del temporizador
Preseleccionado	DINT	valor inmediato	la duración del retardo (tiempo acumulado)
Acumulador	DINT	valor inmediato	el número de mseg durante el cual el temporizador ha contado el valor inicial es típicamente 0

Estructura TIMER:

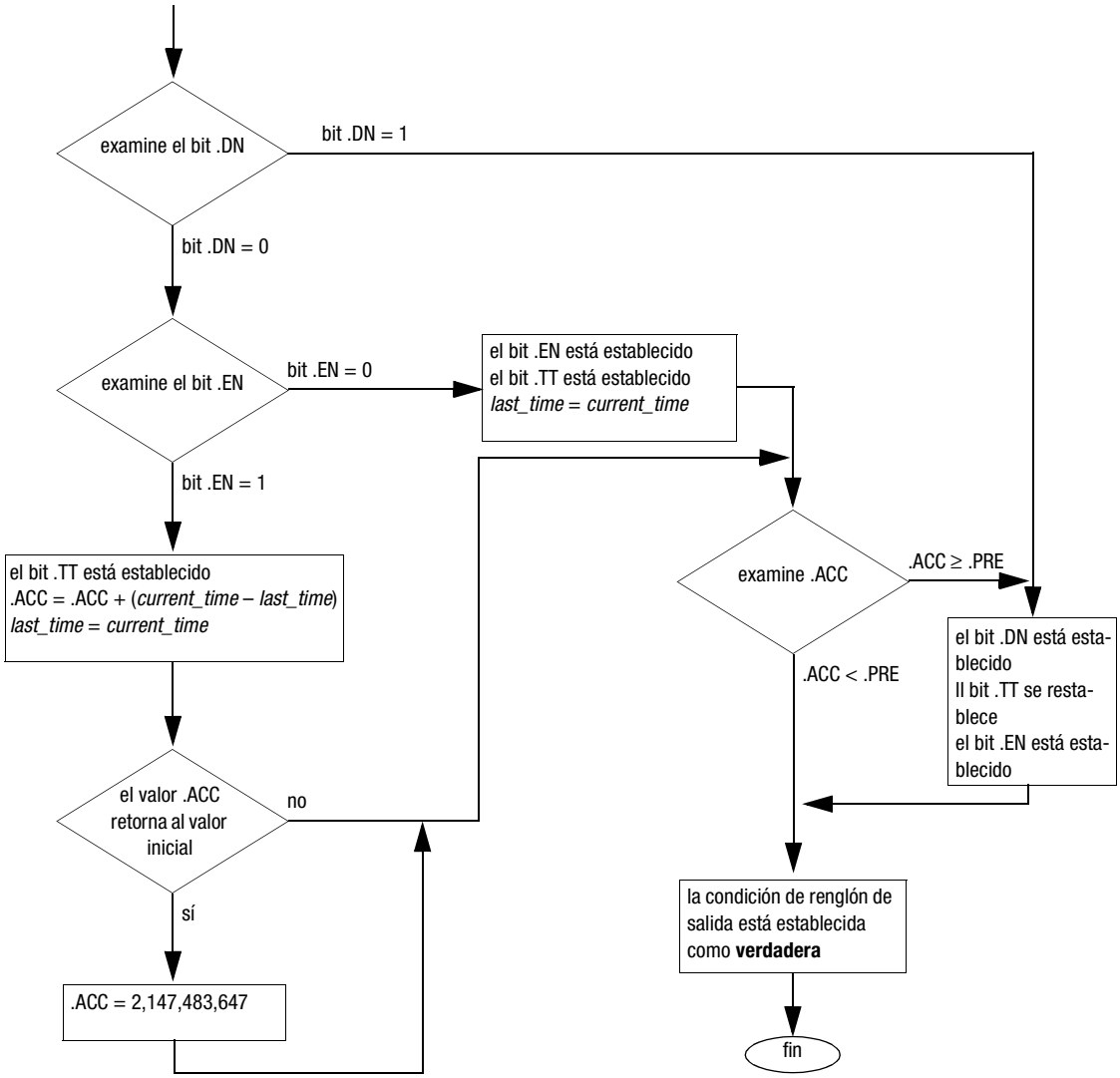
Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción RTO está habilitada.
.TT	BOOL	El bit de temporización indica que hay una operación de temporización en progreso.
.DN	BOOL	El bit de efectuado indica que .ACC ≥ .PRE.
.PRE	DINT	El valor preseleccionado especifica el valor (unidades de 1 mseg) que el acumulador debe alcanzar antes de que la instrucción establezca el bit .DN.
.ACC	DINT	El valor acumulado especifica el número de milisegundos que han transcurrido desde la habilitación de la instrucción RTO.

Descripción: La instrucción RTO es un temporizador retentivo que acumula el tiempo cuando la instrucción está habilitada. La base de tiempo siempre es 1 mseg. Por ejemplo, para un temporizador de 2 segundos, introduzca 2000 para el valor .PRE.

Una vez habilitada, la instrucción RTO acumula el tiempo hasta que se inhabilita. Cuando la instrucción RTO se inhabilita, retiene su valor .ACC. Usted debe restablecer el valor .ACC, típicamente con una instrucción RES que hace referencia a la misma estructura TIMER.


condición de entrada de renglón
bit de habilitación del temporizador (.EN)
condición de renglón que controla la instrucción RES
bit de temporización del temporizador (.TT)
bit de efectuado del temporizador (.DN)
valor preseleccionado
valor acumulado del temporizador (.ACC)
16651
0
el temporizador no llegó al valor .PRE

Ejecución:

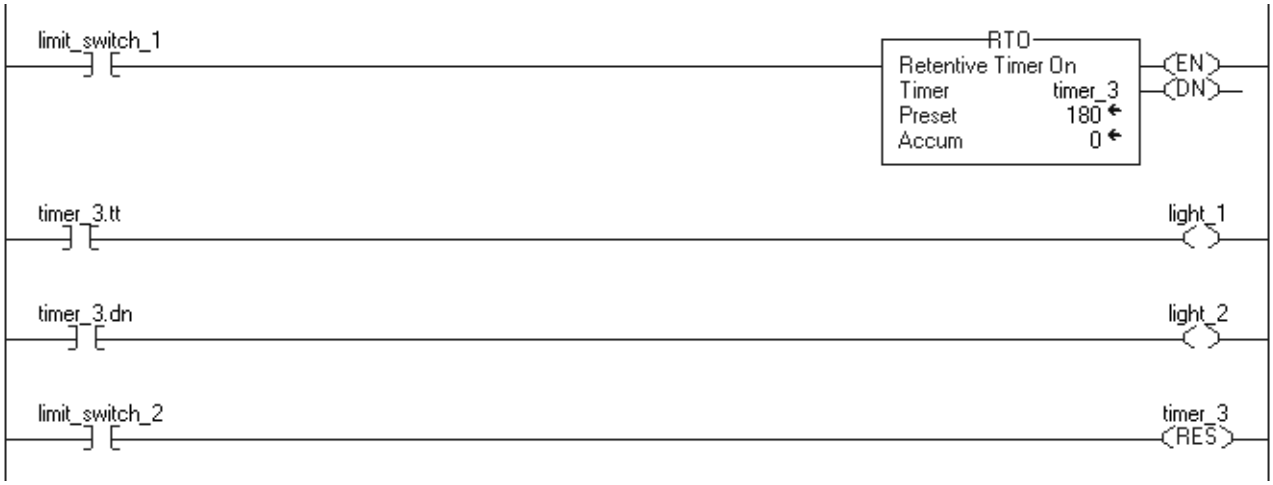
Condición:	Acción:
preescán	El bit .EN se restablece. El bit .TT se restablece. El bit .DN se restablece. El valor .ACC no se modifica. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	El bit .EN se restablece. El bit .TT se restablece. El bit .DN no se modifica. El valor .ACC no se modifica. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es verdadera	

Indicadores de estado aritmético: no afectado

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.PRE < 0	4	34
.ACC < 0	4	34

Ejemplo de RTO:



Cuando *limit_switch_1* se establece, *light_1* está activado durante 180 mseg (*timer_2* está temporizando). Cuando *timer_3.acc* llega a 180, *light_1* se desactiva y *light_2* se activa. *Light_2* permanece activado hasta que *timer_3* se restablece. Si *limit_switch_2* se restablece mientras *timer_3* temporiza, *light_1* permanece activado. Cuando *limit_switch_2* está establecido, la instrucción RES restablece *timer_3* (restablece los bits de estado y el valor .ACC).

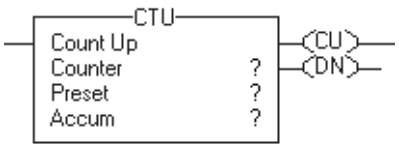
Otros formatos:

Formato:	Sintaxis:
texto neutro	`RTO(timer,preset,accum);`
texto ASCII	`RTO timer preset accum`

Instrucciones relacionadas: TON, TOF, RES

Count Up (CTU)

La instrucción CTU es una instrucción de salida.



Operandos:

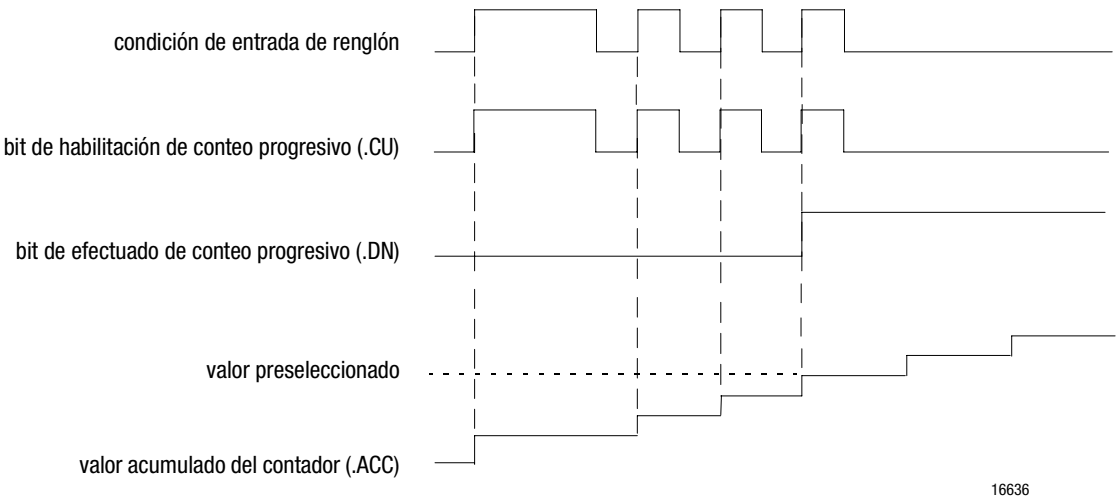
Operando:	Tipo:	Formato:	Descripción:
Counter	COUNTER	tag	estructura del contador
Preseleccio-nado	DINT	valor inme-diato	el conteo máximo
Acumulador	DINT	valor inme-diato	el número de veces que el tem-porizador ha contado el valor inicial es típicamente 0

estructura COUNTER

Mnemónico:	Tipo de datos:	Descripción:
.CU	BOOL	El bit de habilitación de conteo progresivo indica que la instrucción CTU está habilitada.
.DN	BOOL	El bit de efectuado indica que .ACC ≥ .PRE.
.OV	BOOL	El bit de overflow indica que el contador excedió el límite superior de 2,147,483,647. El contador llega a –2,147,483,648 y retorna al valor inicial.
.UN	BOOL	El bit de underflow indica que el contador excedió el límite superior de –2,147,483,648. El contador llega a 2,147,483,647 y vuelve a contar regresivamente.
.PRE	DINT	El valor preseleccionado especifica el valor al cual acumulador debe llegar antes de que la instrucción establezca el bit .DN.
.ACC	DINT	El valor acumulado especifica el número de transiciones que la instrucción ha contado.

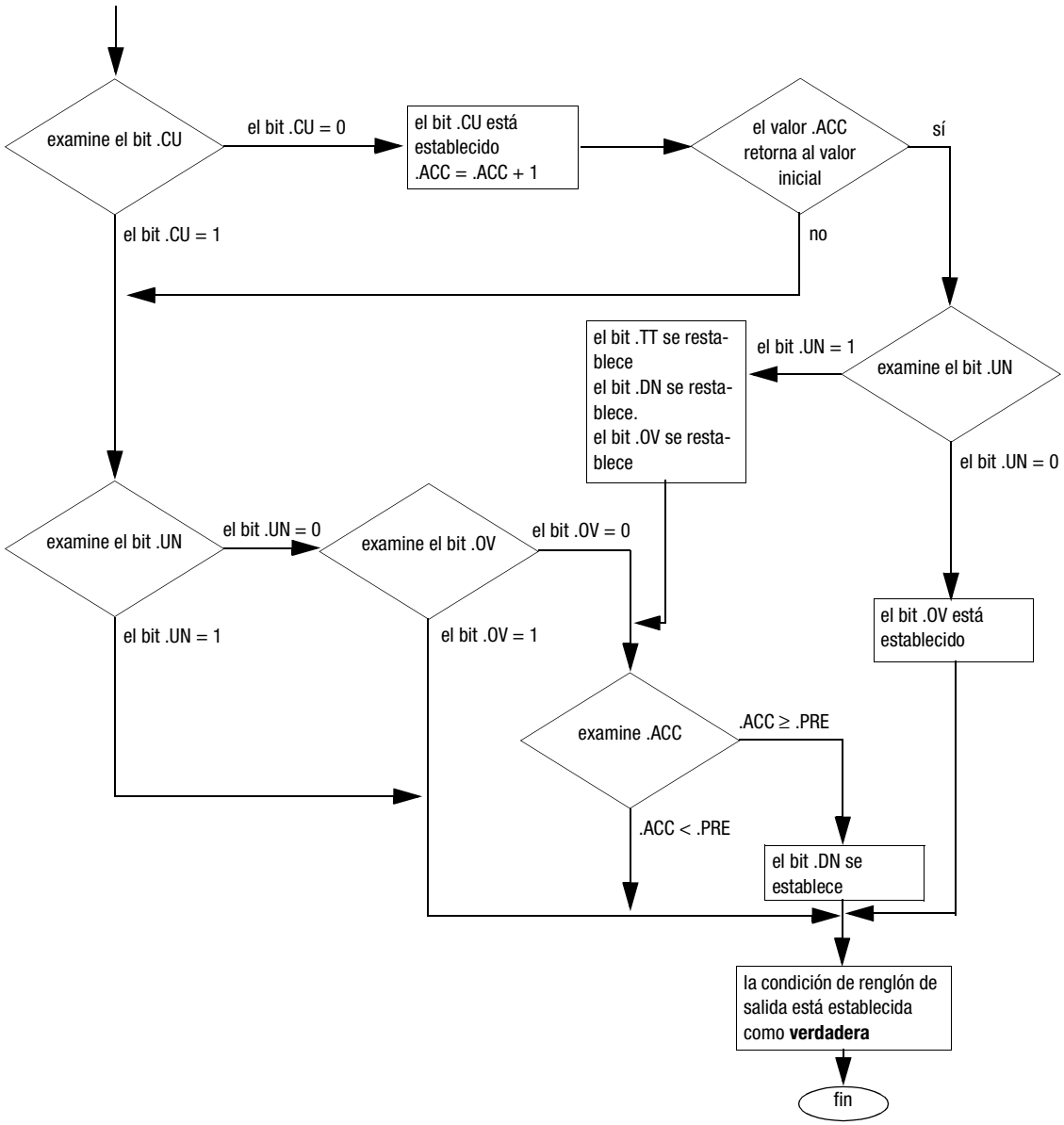
Descripción: La instrucción CTU cuenta progresivamente.

Una vez habilitada y cuando el bit .CU está restablecido, la instrucción CTU incrementa el contador por uno. Una vez habilitada y el bit .CU está establecido, o una vez inhabilitada, la instrucción CTU retiene su valor .ACC.


condición de entrada de renglón
bit de habilitación de conteo progresivo (.CU)
bit de efectuado de conteo progresivo (.DN)
valor preseleccionado
valor acumulado del contador (.ACC)
16636

El valor acumulador continúa incrementándose incluso después del establecimiento del bit .DN. Para restablecer el valor acumulado, use una instrucción RES que haga referencia a la estructura del contador o escriba 0 al valor acumulado.

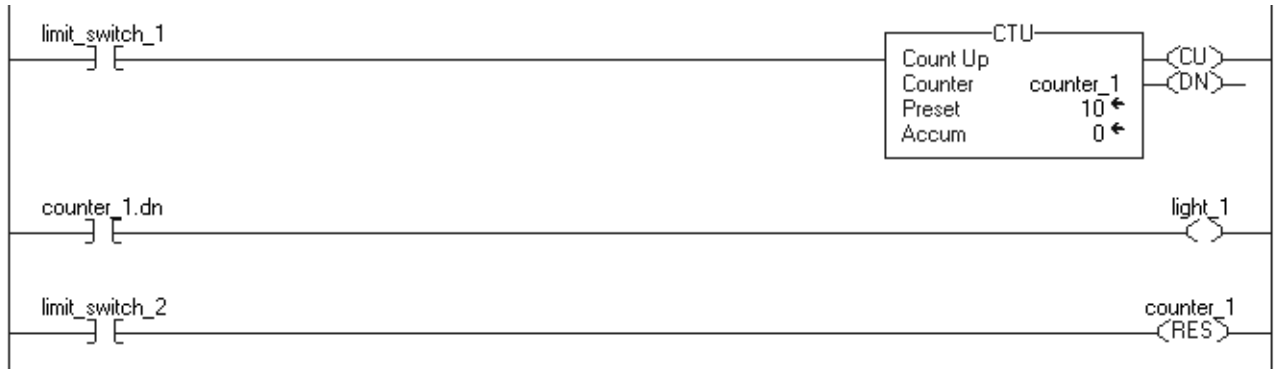
Ejecución:

Condición:	Acción:
preescán	El bit .CU se establece para evitar los incrementos no válidos durante el primer escán del programa. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	El bit .CU se restablece. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es verdadera	

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de CTU:



Después que *limit_switch_1* cambia de inhabilitado a habilitado 10 veces, el bit .DN se establece y *light_1* se activa. Si *limit_switch_1* continúa cambiando de inhabilitado a habilitado, *counter_1* continúa incrementando el conteo y el bit .DN permanece establecido. Cuando limit_switch_2 está habilitado, la instrucción RES restablece *counter_1* (restablece los bits de estado y el valor .ACC) y *light_1* se desactiva.

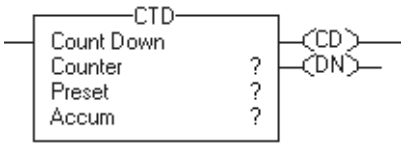
Otros formatos:

Formato:	Sintaxis:
texto neutro	`CTU(counter,preset,accum);`
texto ASCII	`CTU counter preset accum`

Instrucciones relacionadas: CTD, RES

Count Down (CTD)

La instrucción CTD es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Counter	COUNTER	tag	estructura del contador
Preseleccio-nado	DINT	valor inme-diato	el conteo mínimo
Acumulador	DINT	valor inme-diato	el número de veces que el tem-porizador ha contado el valor inicial es típicamente 0

estructura COUNTER

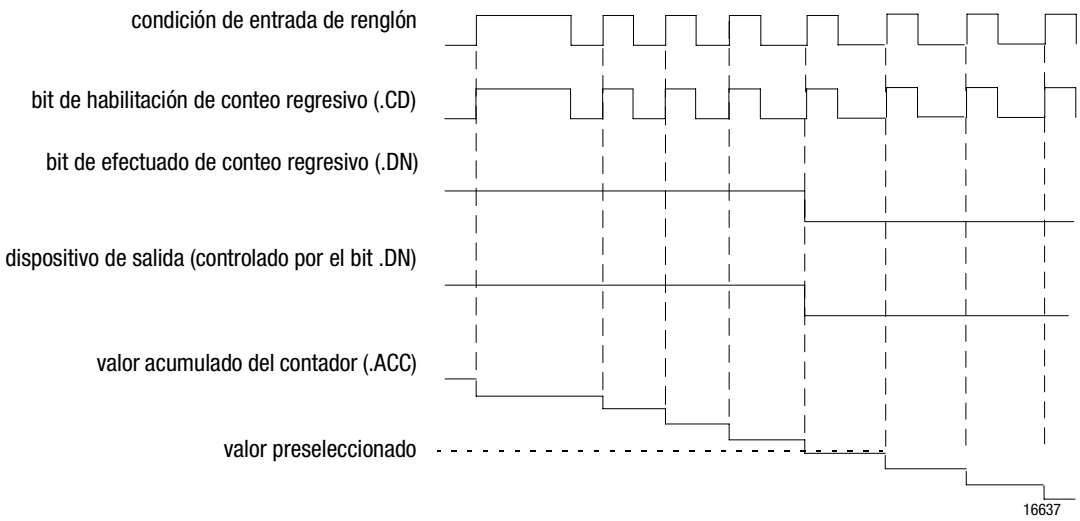
Mnemónico:	Tipo de datos:	Descripción:
.CD	BOOL	El bit de habilitación de conteo regresivo indica que la instrucción CTD está habilitada.
.DN	BOOL	El bit de efectuado indica que .ACC ≥ .PRE.
.OV	BOOL	El bit de overflow indica que el contador excedió el límite superior de 2,147,483,647. El contador llega a –2,147,483,648 y retorna al valor inicial.
.UN	BOOL	El bit de underflow indica que el contador excedió el límite superior de –2,147,483,648. El contador llega a 2,147,483,647 y vuelve a contar regresivamente.
.PRE	DINT	El valor preseleccionado especifica el valor al cual el acumulador debe llegar antes de que la instrucción establezca el bit .DN.
.ACC	DINT	El valor acumulado especifica el número de transiciones que la instrucción ha contado.

Descripción:

La instrucción CTD cuenta regresivamente.

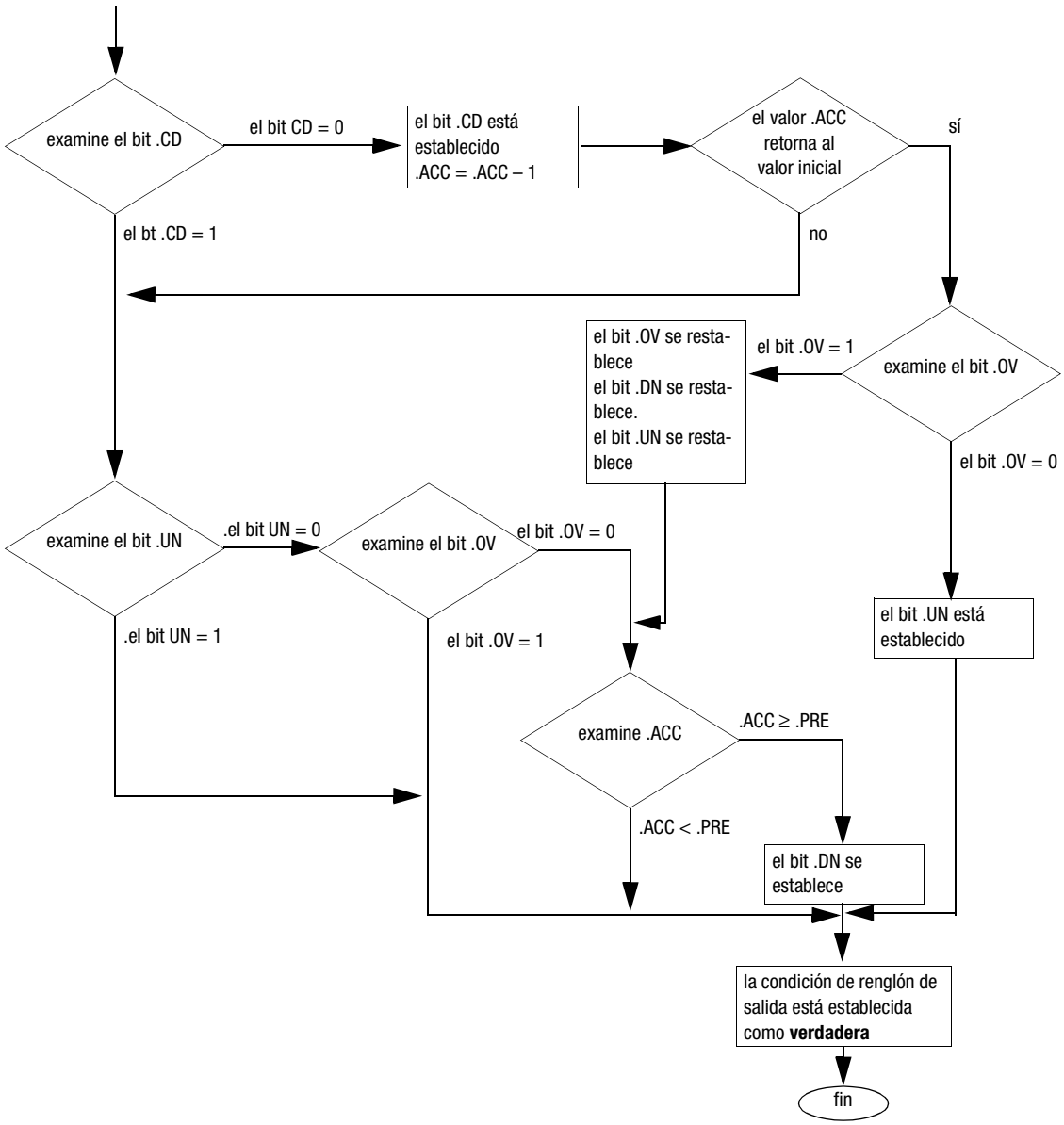
La instrucción CTD se usa típicamente con una instrucción CTU que hace referencia a la misma estructura del contador.

Una vez habilitada y cuando el bit .CD está restablecido, la instrucción CTD decrementa el contador por uno. Una vez habilitada y el bit .CD está establecido, o una vez inhabilitada, la instrucción CTD retiene su valor .ACC.


condición de entrada de renglón
bit de habilitación de conteo regresivo (.CD)
bit de efectuado de conteo regresivo (.DN)
dispositivo de salida (controlado por el bit .DN)
valor acumulado del contador (.ACC)
valor preseleccionado
16637

El valor acumulador continúa decrementándose incluso después del establecimiento del bit .DN. Para restablecer el valor acumulado, use una instrucción RES que haga referencia a la estructura del contador o escriba 0 al valor acumulado.

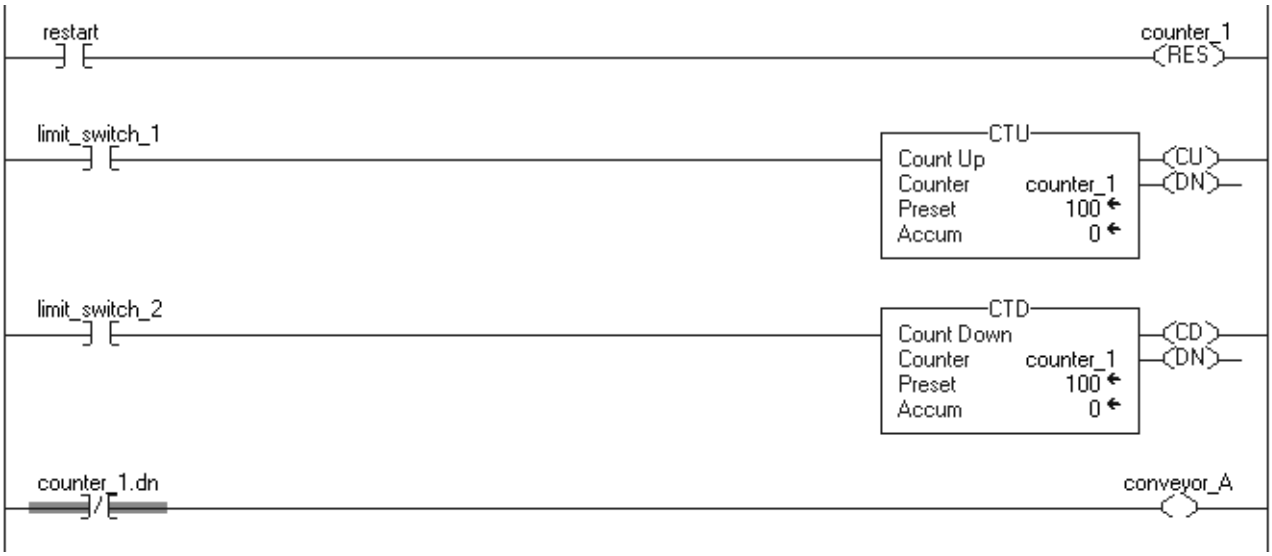
Ejecución:

Condición:	Acción:
preescán	El bit .CD se establece para evitar los decrementos no válidos durante el primer escán del programa. La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	El bit .CD se restablece. La condición de salida de salida se establece como falsa.
la condición de salida de renglón es verdadera	

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de CTD:



Un transportador mueve piezas en una zona de búfer. Cada vez que entra una pieza, *limit_switch_1* se habilita y *counter_1* se incrementa por 1. Cada vez que una pieza sale, *limit_switch_2* se habilita y *counter_1* se decrementa por 1. Si hay 100 piezas en la zona de búfer (*counter_1.dn* is set), *conveyor_a* se activa e impide que el transportador mueva otras piezas hasta que el búfer cuente con espacio suficiente para más piezas.

Otros formatos:

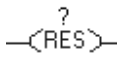
Formato:	Sintaxis:
texto neutro	`CTD(counter,preset,accum);`
texto ASCII	`CTD counter preset accum`

Instrucciones relacionadas: CTU, RES

Reset (RES)

La instrucción RES es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
estructura	TIMER CONTROL COUNTER	tag	estructura para el restablecimiento

Descripción: La instrucción RES restablece una estructura TIMER, COUNTER o CONTROL.

Una vez habilitada, la instrucción RES elimina estos elementos.

Cuando se usa una instrucción RES para:	La instrucción elimina:
temporizador	el valor .ACC los bits de estado de control
contador	el valor .ACC los bits de estado de control
control	el valor .POS los bits de estado de control

ATENCION: Puesto que la instrucción RES elimina el valor .ACC, el bit .DN y el bit .TT, no use la instrucción RES para restablecer un temporizador TOF.

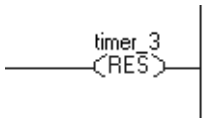
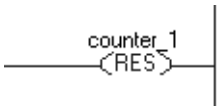
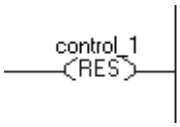
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de salida de renglón es verdadera	La instrucción RES restablece la estructura especificada. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de RES:

Ejemplo:	Descripción:
	Cuando se habilite, restablezca *timer_3*.
	Cuando se habilite, restablezca *counter_1*.
	Cuando se habilite, restablezca *control_1*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`RES(structure);`
texto ASCII	`RES structure`

Notas:

Instrucciones de entrada/salida
(MSG, GSV, SSV)

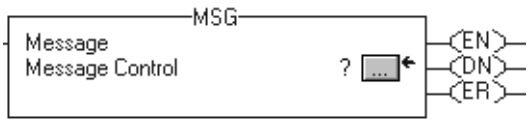
Introducción

Las instrucciones de entrada/salida leen o escriben datos desde o hacia el controlador, o un bloque de datos desde o hacia otro módulo en otra red.

Si usted desea:	Use esta instrucción:	Vea la página:
enviar datos desde o hacia otro módulo	MSG	3-2
obtener información de estado del controlador	GSV	3-27
establecer información de estado del controlador	SSV	3-27

Message (MSG)

La instrucción MSG es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Control de mensajes	Mensaje	tag	estructura del mensaje

Estructura MSG:

Mnemónico:	Tipo de datos:	Descripción:
.FLAGS	INT	El miembro .FLAGS proporciona acceso a los miembros de estado (bits) en una palabra de 16 bits.
		Este bit: **Es este miembro:**
		2 .EW
		4 .ER
		5 .DN
		6 .ST
		7 .EN
		8 .TO
		9 .EN_CC
		Importante: El restablecimiento de cualquiera de los bits de estado MSG cuando está habilitado una MSG puede interrumpir las comunicaciones.
.ERR	INT	Si el bit .ER está establecido, la palabra de código de error identifica los códigos de error para la instrucción MSG.
.EXERR	INT	La palabra de código de error extendida especifica información adicional para algunos códigos de error.
.REQ_LEN	INT	La longitud solicitada especifica cuántas palabras la instrucción de mensaje intentará transferir.
.DN_LEN	INT	La longitud efectuada identifica cuántas palabras se transfirieron con éxito.
.EW	BOOL	El bit de habilitación/espera se establece cuando el controlador detecta que una solicitud de mensaje ha entrado en la cola. El controlador restablece el bit .EW cuando se establece el bit .ST.
.ER	BOOL	Se establece el bit de error cuando el controlador detecta el fallo de una transferencia. El bit .ER se restablece la próxima vez que la condición de renglón de entrada va de falsa a verdadera.
.DN	BOOL	Se establece el bit de efectuado cuando se transfiere con éxito el último paquete del mensaje. El bit .DN se restablece la próxima vez que la condición de renglón de entrada va de falsa a verdadera.
.ST	BOOL	Se establece el bit de arranque cuando el controlador comienza a ejecutar la instrucción MSG. El bit .ST se restablece cuando se establece el bit .DN o .ER.

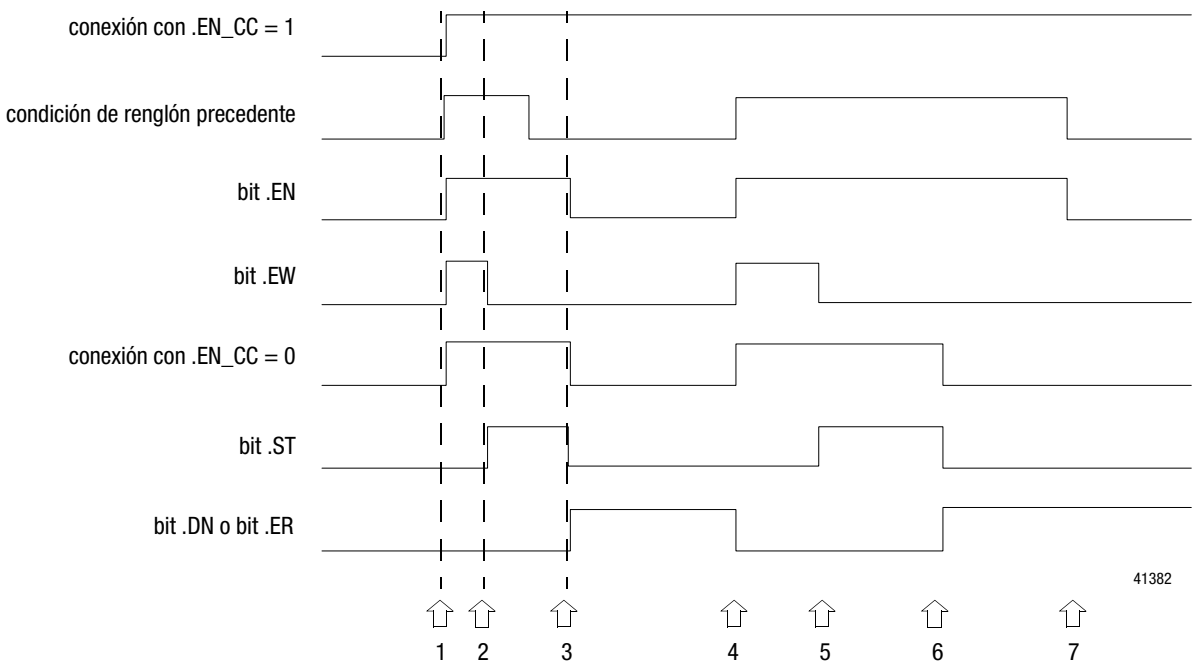
Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	Se establece el bit de habilitación cuando la condición de renglón de entrada se hace verdadera y permanece establecida hasta que se establece el bit .DN o .ER y la condición de renglón de entrada es falsa. Si la condición de entrada de renglón se hace falsa, pero los bits .DN y .ER se restablecen, el bit .EN permanece establecido.
.TO	BOOL	Si se establece manualmente el bit .TO, el controlador interrumpe el procesamiento del mensaje y establece el bit .ER.
.EN_CC	BOOL	El bit de habilitación de caché determina cómo administrar la conexión MSG. Consulte “Cómo seleccionar una opción de caché:” en la pagina 3-26. Las conexiones para las instrucciones MSG que salen del puerto en serie no se ponen en caché aun cuando el bit .EN_CC está establecido.

ATENCION: El controlador procesa los bits .ST y .EW asíncronamente con el escán del programa. Para examinar estos bits en la lógica de escalera, copie la palabra .FLAGS a un tag INT y verifique los bits en ese lugar. De lo contrario, los problemas de temporización pueden hacer no válida su aplicación y resultar en la posibilidad de daños del equipo y lesiones personales.

Descripción: La instrucción MSG lee o escribe asíncronamente un bloque de datos a otro módulo en una red.

La instrucción MSG transfiere los elementos de datos. El tamaño de cada elemento depende de los tipos de datos que se especifican y el tipo de comando de mensaje que se usa.

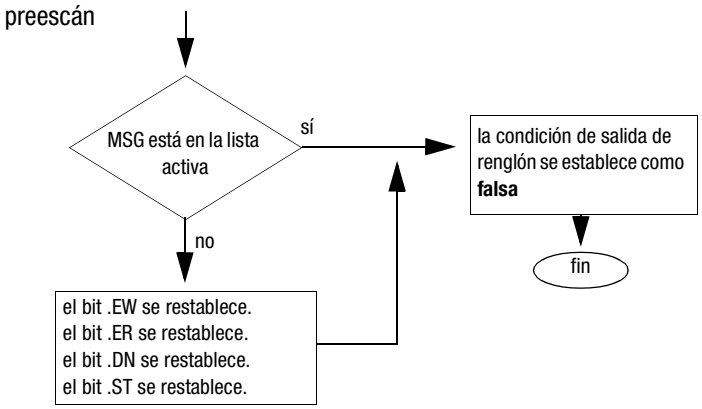
Diagrama de temporización de la instrucción MSG



donde:	Descripción:
1	la condición de renglón precedente es verdadera el bit .EN está establecido el bit .EW está establecido la conexión está abierta
2	el mensaje ha sido enviado el bit .ST está establecido el bit .EW se restablece
3	el mensaje se ha efectuado o ha entrado en error y una condición de entrada de renglón es falsa el bit .DN o .ER está establecido el bit .ST se restablece la conexión está cerrada (si .EN_CC = 0) el bit .EN se restablece (debido a que la condición de entrada de renglón es falsa)
4	la condición de entrada de renglón es verdadera y el bit .DN o .ER se establece previamente el bit .EN está establecido el bit .EW está establecido la conexión está abierta el bit .DN o .ER está restablecido
5	el mensaje ha sido enviado el bit .ST está establecido el bit .EW se restablece
6	el mensaje se ha efectuado o ha entrado en error y una condición de entrada de renglón todavía es verdadera el bit .DN o .ER está establecido el bit .ST se restablece la conexión está cerrada (si .EN_CC = 0)
7	la condición de de entrada renglón se hace verdadera y el bit .DN o .ER se establece el bit .EN se restablece

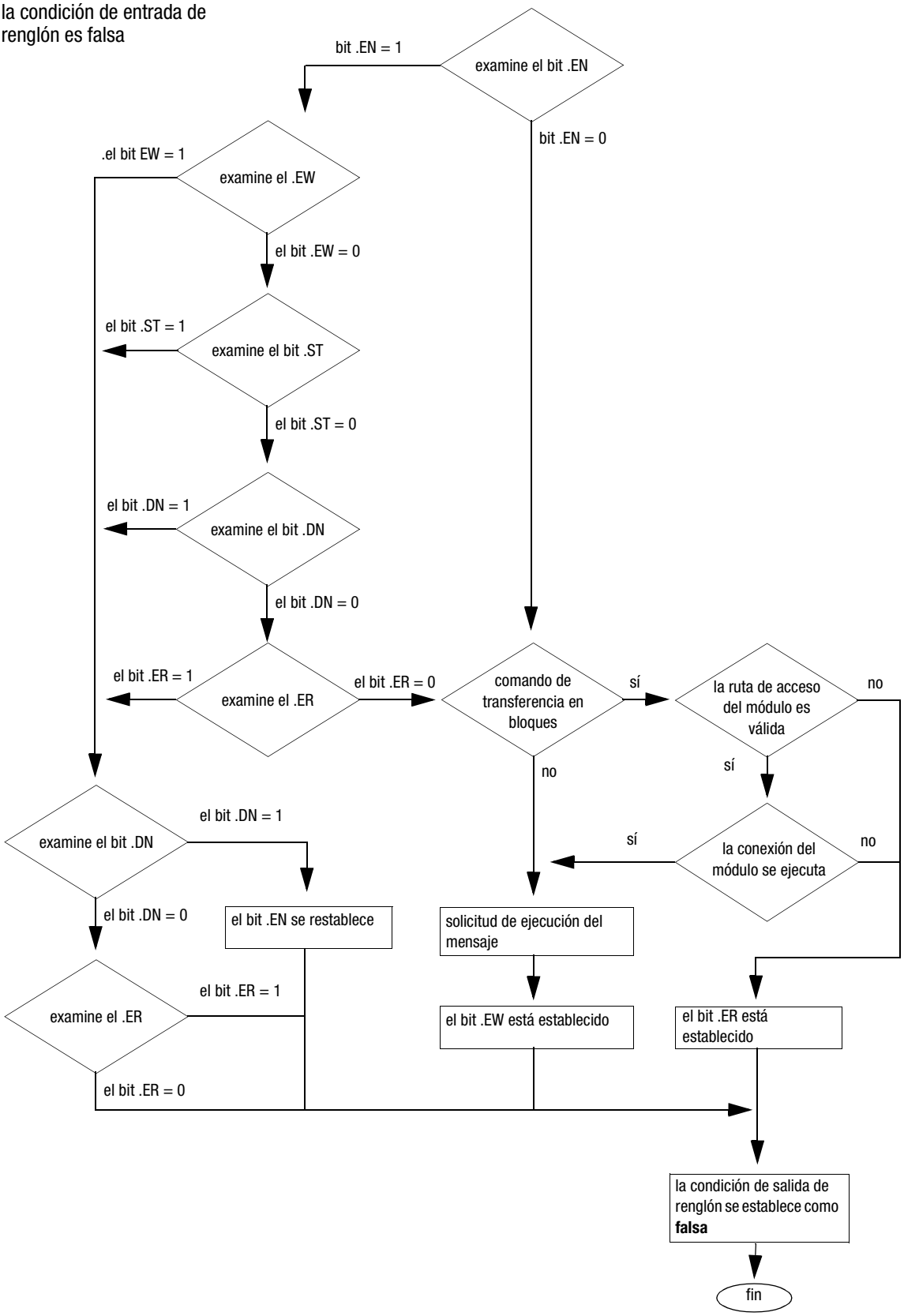
Ejecución:

Condición:	Acción:

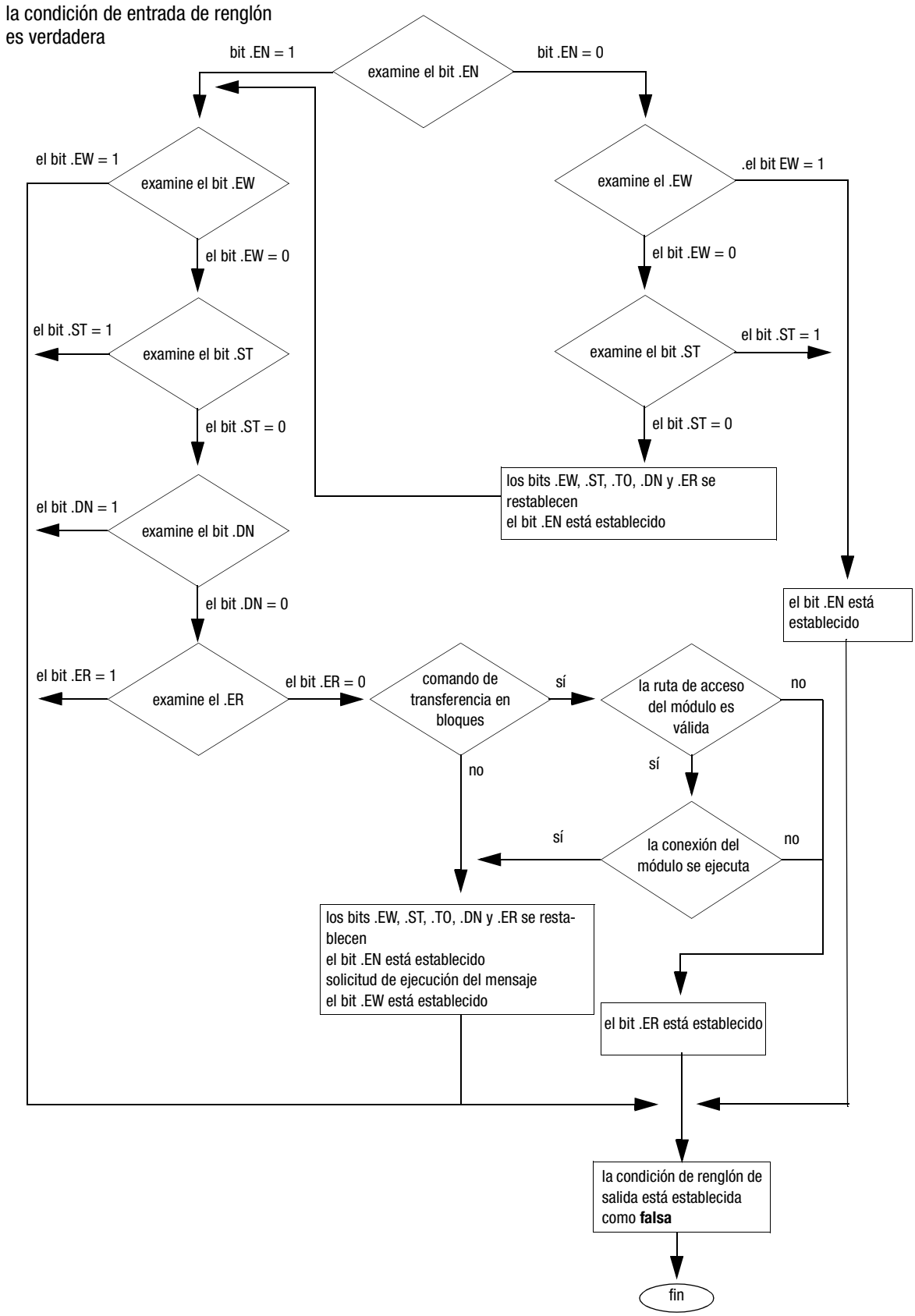

preescán
MSG está en la lista activa
sí
no
la condición de salida de renglón se establece como **falsa**
fin
el bit .EW se restablece.
el bit .ER se restablece.
el bit .DN se restablece.
el bit .ST se restablece.

Condición: **Acción:**

la condición de entrada de renglón es falsa


examine el bit .EN
bit .EN = 1
bit .EN = 0
examine el .EW
.el bit EW = 1
el bit .EW = 0
examine el bit .ST
el bit .ST = 1
el bit .ST = 0
examine el bit .DN
el bit .DN = 1
el bit .DN = 0
examine el .ER
el bit .ER = 1
el bit .ER = 0
comando de transferencia en bloques
sí
no
la ruta de acceso del módulo es válida
sí
no
la conexión del módulo se ejecuta
sí
no
examine el bit .DN
el bit .DN = 1
el bit .DN = 0
el bit .EN se restablece
solicitud de ejecución del mensaje
examine el .ER
el bit .ER = 1
el bit .ER = 0
el bit .EW está establecido
el bit .ER está establecido
la condición de salida de renglón se establece como **falsa**
fin

Condición: **Acción:**


la condición de entrada de renglón es verdadera
examine el bit .EN
bit .EN = 1
bit .EN = 0
examine el bit .EW
el bit .EW = 1
el bit .EW = 0
examine el .EW
.el bit EW = 1
el bit .EW = 0
examine el bit .ST
el bit .ST = 1
el bit .ST = 0
examine el bit .ST
el bit .ST = 1
el bit .ST = 0
los bits .EW, .ST, .TO, .DN y .ER se restablecen
el bit .EN está establecido
examine el bit .DN
el bit .DN = 1
el bit .DN = 0
el bit .EN está establecido
examine el .ER
el bit .ER = 1
el bit .ER = 0
comando de transferencia en bloques
sí
la ruta de acceso del módulo es válida
no
sí
no
la conexión del módulo se ejecuta
sí
no
los bits .EW, .ST, .TO, .DN y .ER se restablecen
el bit .EN está establecido
solicitud de ejecución del mensaje
el bit .EW está establecido
el bit .ER está establecido
la condición de renglón de salida está establecida como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Otros formatos:

Formato:	Sintaxis:
texto neutro	`MSG(message_control);`
texto ASCII	`MSG(message_control)`

MSG Error Codes

Los códigos de error dependen del tipo de la instrucción MSG.

Códigos de error ControlLogix (CIP)

El software de programación no siempre muestra en pantalla la amplia descripción completa de los códigos de error ControlLogix (CIP).

Código de error (hex):	Descripción:	Cómo se muestra en el software:
0001	Fallo de conexión (vea los códigos de error extendidos)	idéntico a la descripción
0002	Recurso no suficiente	idéntico a la descripción
0003	Valor no válido	idéntico a la descripción
0004	Error de sintaxis IOI8 (vea los códigos de error extendidos)	idéntico a la descripción
0005	Destino desconocido, clase no compatible, ocurrencia no definida o elemento de estructura no definido (vea los códigos de error extendidos)	idéntico a la descripción
0006	Espacio de paquete no suficiente	idéntico a la descripción
0007	Conexión perdida	idéntico a la descripción
0008	Servicio no compatible	idéntico a la descripción
0009	Error en el segmento de datos o valor de atributo no válido	idéntico a la descripción
000A	Error de la lista de atributos	idéntico a la descripción
000B	El estado ya existe	idéntico a la descripción
000C	Conflicto con el modelo de objeto	idéntico a la descripción
000D	El objeto ya existe	idéntico a la descripción
000E	El atributo no se puede guardar	idéntico a la descripción
000F	Permiso rechazado	idéntico a la descripción
0010	Conflicto con el estado del dispositivo	idéntico a la descripción
0011	No hay espacio suficiente para la respuesta	idéntico a la descripción
0012	El fragmento es primitivo	idéntico a la descripción
0013	No hay datos de comando suficientes	idéntico a la descripción
0014	El atributo no es compatible	idéntico a la descripción

Código de error (hex):	Descripción:	Cómo se muestra en el software:
0015	Exceso de datos	idéntico a la descripción
001A	La solicitud de puente es demasiado grande	idéntico a la descripción
001B	La respuesta de puente es demasiado grande	idéntico a la descripción
001C	Pocos atributos en la lista de atributos	idéntico a la descripción
001D	Lista de atributos no válida	idéntico a la descripción
001E	Error de servicio incorporado	idéntico a la descripción
001F	Fallo asociado con la conexión (vea los códigos de error extendidos)	idéntico a la descripción
0022	Se recibió respuesta no válida	idéntico a la descripción
0025	Error de segmento clave	idéntico a la descripción
0026	Error IOI no válido	idéntico a la descripción
0027	Atributo no esperado en la lista	idéntico a la descripción
0028	Error DeviceNet – identificación de miembro no válida	idéntico a la descripción
0029	Error DeviceNet – el miembro no se puede establecer	idéntico a la descripción

Códigos de error extendidos ControlLogix

Estos son los códigos de error extendidos ControlLogix (CIP). El software no muestra en pantalla el texto para los códigos de error extendidos. Estos son los códigos de error extendidos para el código de error **0001**.

Código de error extendido (hex):	Descripción:
0100	Conexión en uso
0103	Transporte incompatible
0106	Conflicto de propiedad
0107	No se encontró la conexión
0108	Tipo de conexión no válido
0109	Tamaño de conexión no válido
0110	Módulo no configurado
0111	EPR no es compatible
0114	Módulo equivocado
0115	Tipo de dispositivo equivocado
0116	Revisión equivocada
0118	Formato de configuración no válido
011A	Aplicación sin conexiones

Código de error extendido (hex):	Descripción:
0203	Tiempo de espera de la conexión
0204	Tiempo de espera de mensaje no conectado
0205	Error de parámetro de envío no conectado
0206	Mensaje demasiado grande
0301	No hay memoria de búfer
0302	Ancho de banda no disponible
0303	No hay filtros disponibles
0305	Coincidencia de firma
0311	Puerto no disponible
0312	Dirección de vínculo no disponible
0315	Tipo de segmento no válido
0317	Conexión no programada

Estos son los códigos de error extendidos para el código de error **001F**.

Código de error extendido (hex):	Descripción:
0203	Tiempo de espera de la conexión

Estos son los códigos de error extendidos para el código de error **0004** y **0005**.

Código de error extendido (hex):	Descripción:
0000	el estado extendido ya no tiene memoria
0001	el estado extendido ya no tiene ocurrencias

Códigos de error PLC y SLC (.ERR)

El software de programación no muestra en pantalla la descripción completa de los códigos de error PLC y SLC.

Código de error (hex):	Descripción:	Cómo se muestra en el software:
0010	Comando o formato no válido del procesador local	Conflicto con el estado del dispositivo
0020	El módulo de comunicación no funciona	Error desconocido
0030	El nodo remoto falta, está desconectado o desactivado	Error desconocido
0040	El procesador está conectado pero entró en fallo (hardware)	Error desconocido
0050	Número de estación equivocado	Error desconocido
0060	La función solicitada no está disponible	Error desconocido
0070	El procesador está en el modo de programación	Error desconocido
0080	El archivo de compatibilidad del procesador no existe	Error desconocido
0090	El nodo remoto no puede almacenar en un búfer el comando	Error desconocido
00B0	El procesador está descargando y por lo tanto no hay acceso al mismo	Error desconocido
00F0	Error PCCC (vea los códigos de error extendidos)	Error desconocido

Códigos de error extendidos PLC y SLC (.EXERR)

El software no muestra en pantalla el texto para los códigos de error extendidos. Estos son los códigos de error extendidos para el código de error **00F0**.

Código de error extendido (hex):	Descripción:
0001	El procesador convirtió incorrectamente la dirección
0002	Dirección no completa
0003	Dirección incorrecta
0004	Formato de dirección no válido – no se encuentra el símbolo
0005	Formato de dirección no válido – el símbolo tiene 0 ó mayor que el número máximo de caracteres compatibles con el dispositivo
0006	El archivo de dirección no existe en el procesador receptor
0007	El archivo de destino es demasiado pequeño para el número de palabras solicitadas
0008	No se puede realizar la solicitud La situación cambió durante la operación de multipaquetes
0009	Los datos o el archivo son demasiado grandes No hay memoria disponible
000A	El procesador receptor no puede colocar la información solicitada en paquetes
000B	Error de privilegio; se rechazó el acceso
000C	La función solicitada no está disponible
000D	La solicitud es redundante
000E	El comando no se puede ejecutar
000F	Overflow; overflow de histograma

Código de error extendido (hex):	Descripción:
0010	No hay acceso
0011	El tipo de datos solicitado no coincide con los datos disponibles
0012	Parámetros de comando incorrectos
0013	Referencia de dirección existe a la zona eliminada
0014	Fallo de ejecución de comando debido a condición desconocida Overflow de histograma PLC-3
0015	Error de conversión de datos
0016	El escáner no está disponible para comunicarse con un adaptador de rack 1771
0017	El adaptador no está disponible para comunicarse con el módulo
0018	La respuesta del módulo 1771 no es válida
0019	Etiqueta duplicada
001A	Propietario del archivo activo – se usa el archivo
001B	Propietario del programa activo – se descarga o se edita en línea
001C	El disco está protegido contra la escritura o no ofrece acceso (fuera de línea solamente)
001D	Otra aplicación está usando el archivo de disco No se realizó la actualización (fuera de línea solamente)

Códigos de error de transferencia en bloques

Estos son los códigos de error específicos para la transferencia en bloques ControlLogix.

Código de error (hex):	Descripción:	Cómo se muestra en el software:
00D0	El escáner no recibió una respuesta de transferencia en bloques del módulo de transferencia en bloques dentro de 3.5 segundos de la solicitud	error desconocido
00D1	La suma de comprobación de la respuesta de lectura no coincidió con la suma de comprobación de la cadena de datos	error desconocido
00D2	El escáner solicitó una lectura o escritura pero el módulo de transferencia en bloques respondió con lo opuesto	error desconocido
00D3	El escáner solicitó una longitud determinada y el módulo de transferencia en bloques respondió con otra longitud	error desconocido
00D6	El escáner recibió una respuesta del módulo de transferencia en bloques indicando que la solicitud de escritura entró en fallo	error desconocido
00EA	El escáner no se configuró para comunicarse con el rack que contendría este módulo de transferencia en bloques	error desconocido
00EB	La ranura lógica especificada no está disponible para el tamaño de rack especificado	error desconocido
00EC	Existe actualmente una solicitud de transferencia en bloques y se requiere una respuesta antes de que otra respuesta se pueda iniciar	error desconocido
00ED	El tamaño de la solicitud de transferencia en bloques no corresponde a las solicitudes de tamaño de transferencia en bloques válidas	error desconocido
00EE	El tipo de solicitud de transferencia en bloques no corresponde a los comandos BT_READ o BT_WRITE esperados	error desconocido
00EF	El escáner no pudo encontrar una ranura disponible en la tabla de transferencias en bloques para aceptar la solicitud de transferencia en bloques	error desconocido
00F0	El escáner recibió una solicitud para restablecer los canales de E/S remotas mientras existían transferencias en bloques pendientes	error desconocido
00F3	Las colas para transferencias en bloques remotas están llenas	error desconocido
00F5	No hay canales de comunicación configurados para el rack o ranura solicitado	error desconocido
00F6	No hay canales de comunicación configurados para las E/S remotas	error desconocido
00F7	Se sobrepasó el tiempo de espera, establecido en la instrucción, de la transferencia en bloques antes de completarse	error desconocido
00F8	Error del protocolo de la transferencia en bloques – transferencia en bloques no solicitada	error desconocido
00F9	Se perdieron datos de transferencia debido a un canal de comunicación deficiente	error desconocido
00FA	El módulo de transferencia en bloques solicitó una longitud diferente que la solicitada por la instrucción de transferencia en bloques asociada	error desconocido
00FB	La suma de comprobación de los datos de transferencias en bloques de lectura no es correcta	error desconocido
00FC	Se realizó una transferencia no válida de los datos de transferencia en bloques de escritura entre el adaptador y el módulo de transferencia en bloques	error desconocido
00FD	El tamaño de la transferencia en bloques más el tamaño del índice en la tabla de datos de transferencia en bloques son mayores que el tamaño del archivo de la tabla de datos de transferencia en bloques	error desconocido

Códigos de error Logix5550

Estos son los códigos de error Logix5550.

Código de error (hex):	Descripción:	Cómo se muestra en el software:
00D0	Instancia de asignación no definida	error desconocido
00D1	El módulo no está en el estado marcha	error desconocido
00FB	Puerto de mensaje no compatible	error desconocido
00FC	Tipo de datos de mensaje no compatible	error desconocido
00FD	Mensaje no inicializado	error desconocido
00FE	Tiempo de espera de mensaje	error desconocido
00FF	Error general (vea los códigos de error extendidos)	error desconocido

Códigos de error extendidos Logix5550

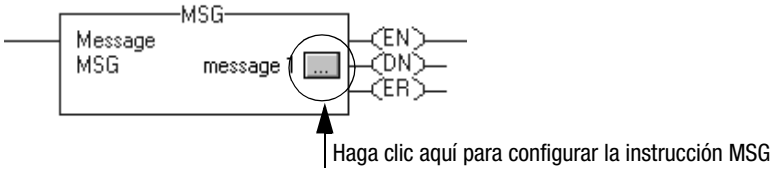
Estos son los códigos de error extendidos Logix5550. El software no muestra en pantalla el texto para los códigos de error extendidos. Estos son los códigos de error extendidos para el código de error **00FF**.

Código de error extendido (hex):	Descripción:
2001	Exceso de IOI
2002	Valor de parámetro incorrecto
2018	Rechazo de semáforo
201B	Tamaño demasiado pequeño
201C	Tamaño no válido
2100	Fallo de privilegio
2101	Posición del interruptor de llave no válida
2102	Contraseña no válida
2103	No se emitió una contraseña
2104	Dirección fuera de rango
2105	Dirección y cantidad fuera de rango
2106	Datos en uso

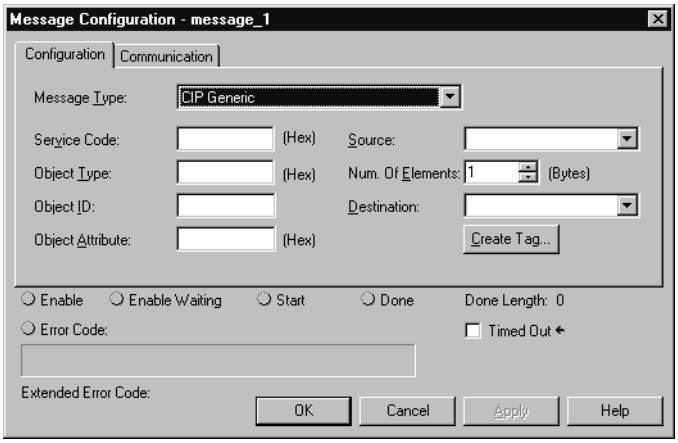
Código de error extendido (hex):	Descripción:
2107	El tipo no es válido o no es compatible
2108	El controlador está en el modo carga o descarga
2109	Intento para cambiar el número de dimensiones de registro
210A	Nombre de símbolo no válido
210B	El símbolo no existe
210E	La búsqueda entró en fallo
210F	La tarea no se puede iniciar
2110	No se puede escribir
2111	No se puede leer
2112	No se puede editar la rutina compartida
2113	El controlador está en el modo fallo
2114	Modo marcha inhibido

Cómo especificar los detalles de configuración (ficha Configuración)

Después de introducir la instrucción MSG y especificar la estructura MESSAGE, use la ficha Configuración del software de programación para especificar los detalles del mensaje.



Los detalles que usted configura dependen del tipo de mensaje que selecciona.



Si el dispositivo receptor es un:	Seleccione uno de estos tipos de mensaje:	Vea la página:
Dispositivo ControlLogix o módulo de E/S 1756	Lectura de la tabla de datos CIP	3-14
	Escritura de la tabla de datos CIP	
	CIP genérico	
Procesador PLC-5	TypedRead PLC5	3-16
	TypedWrite PLC5	
	Lectura de rango de palabra PLC5	
	Escritura de rango de palabra PLC5	
Controlador SLC	Typed Read SLC	3-17
	Typed Write SLC	
Módulo de transferencia en bloques mediante una red de E/S remotas universales	Transferencia en bloques de lectura	3-17
	Transferencia en bloques de escritura	
Procesador PLC-3	TypedRead PLC3	3-18
	TypedWrite PLC-3	
	Lectura de rango de palabra PLC3	
	Escritura de rango de palabra PLC3	
Procesador PLC-2	Lectura no protegida PLC2	3-19
	Escritura no protegida PLC2	

Usted debe especificar esta información de configuración:

En este campo	Especifique:
Elemento de origen/Tag	Si selecciona un tipo de mensaje de lectura, el Elemento de origen es la dirección de los datos que desea leer en el dispositivo receptor. Use la sintaxis del dispositivo receptor. Si selecciona un tipo de mensaje de escritura, el Tag de origen es el tag de los datos en el controlador Logix5550 que desea enviar al dispositivo receptor.
Número de elementos	El número de elementos que lee/escribe depende del tipo de datos que usa. Un elemento se refiere a un segmento de datos asociados. Por ejemplo, el tag *timer1* es un elemento que consiste en una estructura de control del temporizador.
Elemento de destino/Tag	Si selecciona un tipo de mensaje de lectura, el Tag de destino es el tag en el controlador Logix5550 donde desea almacenar los datos que lee desde el dispositivo receptor. Si selecciona un tipo de mensaje de escritura, el Elemento de destino es la dirección del lugar en el dispositivo receptor donde desea escribir los datos.

Si especifica un tag de arreglo Logix5550 para el Origen o el Destino, especifique el nombre del tag de arreglo solamente. No incluya los corchetes ni el subíndice de posición.

Cómo especificar los mensajes CIP

Los tipos de mensaje CIP han sido diseñados para transferir los datos hacia o desde otros dispositivos ControlLogix; por ejemplo, cuando se desea enviar un mensaje de un controlador Logix5550 a otro.

Seleccione este comando:	Si usted desea:
Lectura de la tabla de datos CIP	leer datos de otro controlador. Los tipos de Origen y Destino deben coincidir.
Escritura de la tabla de datos CIP	escribir datos a otro controlador. Los tipos de Origen y Destino deben coincidir.
CIP genérico	configurar un mensaje personalizado para enviar los datos de configuración a un módulo de E/S.

Cómo usar mensajes CIP genérico para restablecer los módulos de E/S

Especifique esta información para crear un mensaje personalizado usando el tipo de mensaje CIP genérico.

Si usted desea:	En este campo	Introduzca:	
Realizar un prueba de impulso de un módulo de salida digital	Código de servicio	4c	
	Tipo de objeto	1e	
	ID del objeto	1	
	Atributo del objeto	deje en blanco	
	Origen	*tag_name* de tipo INT [5]	
		Este arreglo contiene:	
		tag_name[0]	máscara de bit de puntos a probar (pruebe sólo un punto a la vez)
		tag_name[1]	reservado, deje 0
		tag_name[2]	anchura de impulso (céntimos de µsegs, por lo general 20)
		tag_name[3]	retardo de cruce por cero para las E/S ControlLogix (centésimos de µsegs, por lo general 40)
		tag_name[4]	verificar el retardo
	Número de elementos	10	
	Destino	deje en blanco	
Restablecer los fusibles electrónicos en un módulo de salida digital	Código de servicio	4d	
	Tipo de objeto	1e	
	Atributo del objeto	deje en blanco	
	ID del objeto	1	
	Origen	*nombre del tag de tipo DINT* Este tag representa una máscara de bit de los puntos en los cuales se restablecen los fusibles.	
	Número de elementos	4	
	Destino	deje en blanco	
Restablecer los diagnósticos enclavados en un módulo de E/S digitales	Código de servicio	4b	
	Tipo de objeto	para un módulo de entrada	1d
		para un módulo de salida	1e
	Atributo del objeto	deje en blanco	
	ID del objeto	1	
	Origen	*tag_name* de tipo DINT Este tag representa una máscara de bit de los puntos en los cuales se restablecen los diagnósticos.	
	Número de elementos	4	
	Destino	deje en blanco	

Si usted desea:	En este campo	Introduzca:
Restablecer el estado enclavado en un módulo analógico	Código de servicio	4b
	Tipo de objeto	a
	Atributo del objeto	introduzca un código para el atributo deseado
	ID del objeto	0
	Origen	deje en blanco
	Número de elementos	0
	Destino	deje en blanco

Cómo especificar los mensajes PLC-5

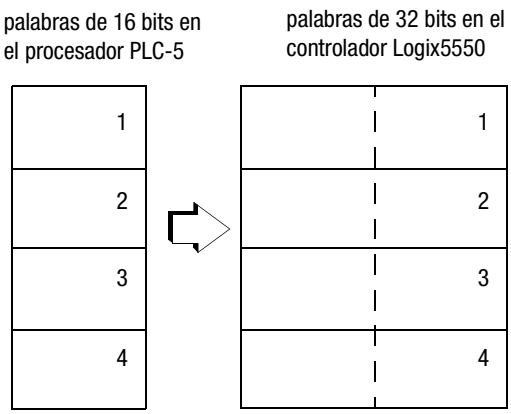
Los tipos de mensaje PLC-5 han sido diseñados para los procesadores PLC-5.

Seleccione este comando:	Si usted desea:
TypedRead PLC5	leer números enteros o datos de tipo REAL. Para los números enteros, este comando lee los números enteros de 16 bits del procesador PLC-5 (tipos de archivo S, B y N) y los almacena en arreglos de datos SINT, INT o DINT en el controlador Logix5550 y mantiene la integridad de los datos. Este comando también lee datos de punto flotante (coma flotante) del procesador PLC-5 (tipo de archivo F) y los almacena en un tag de tipo de datos REAL en el controlador Logix5550.
PLC5 Typed Write	escribir números enteros o datos de tipo REAL. Este comando escribe los datos SINT o INT al procesador PLC-5 (tipos de archivo S, B y N) y mantiene la integridad de datos. Puede escribir los datos DINT siempre que quepan dentro de un tipo de datos INT ($-32{,}768 \geq$ datos $\leq 32{,}767$). Este comando también escribe los datos de tipo REAL del controlador Logix5550 hacia un archivo de punto flotante (coma flotante) PLC-5 (tipo de archivo F).
Lectura de rango de palabra PLC5	leer un rango contiguo de palabras de 16 bits en la memoria PLC-5 independientemente del tipo de datos. Este comando comienza a partir de la dirección especificada como el Elemento de origen y lee secuencialmente el número de palabras de 16 bits solicitadas. Los datos del Elemento de origen se almacenan a partir de la dirección especificada como el Tag de destino.
Escritura de rango de palabra PLC5	escribir un rango contiguo de palabras de 16 bits de la memoria Logix5550 independientemente del tipo de datos hacia la memoria PLC-5. Este comando comienza a partir de la dirección especificada como el tag de origen y lee secuencialmente el número de palabras de 16 bits solicitadas. Los datos del tag de origen se almacenan a partir de la dirección especificada como el Elemento de origen en el procesador PLC-5.

Los comandos TypedRead y TypedWrite también funcionan con los procesadores SLC 5/03 (OS303 y posteriores), procesadores SLC 5/04 (OS402 y posteriores) y procesadores SLC 5/05.

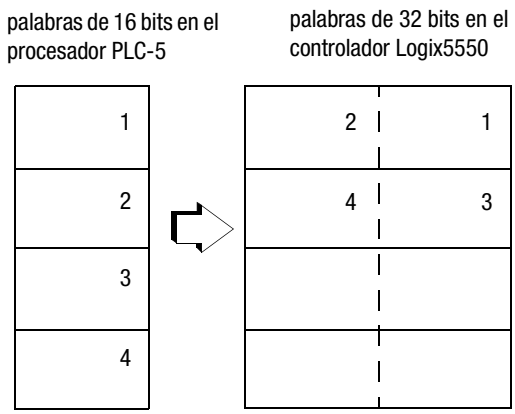
Los diagramas siguientes muestran la diferencias entre los comandos TypedWrite/TypedRead y de rango de palabra. El ejemplo usa los comandos de lectura de un procesador PLC-5 hacia un controlador Logix5550.

Comando TypedRead



Los comandos TypedWrite/TypedRead mantienen la estructura y valor de datos.

Comando de rango de palabra



Los comandos de rango de palabra llenan el tag de destino contiguamente. La estructura y valor de datos se cambian según el tipo de datos de destino.

Cómo especificar los mensajes SLC

Los tipos de mensaje SLC han sido diseñados para los controladores SLC y MicroLogix1000.

Seleccione este comando:	Si usted desea:
Typed Read SLC	leer los datos INT o DINT.
Typed Write SLC	escribir los datos INT o DINT.

El tipo de tag Logix5550 debe coincidir con el tipo de datos SLC. Usted puede transferir solamente los datos INT (que se asignan al tipo de datos de bit SLC) o datos INT (que se asignan al tipo de datos de enteros SLC).

Cómo especificar los mensajes de transferencia en bloques

Los tipos de mensaje de transferencia en bloques se usan para comunicarse con los módulos de transferencia en bloques mediante una red de E/S remotas universales.

Seleccione este comando:	Si usted desea:
Transferencia en bloques de lectura	leer los datos de un módulo de transferencia en bloques. Este tipo de mensaje reemplaza la instrucción BTR.
Transferencia en bloques de escritura	escribir los datos a un módulo de transferencia en bloques. Este tipo de mensaje reemplaza la instrucción BTW.

Los tags de origen (para BTW) y destino (para BTR) deben ser bastante grandes para aceptar los datos solicitados, excepto las estructuras MESSAGE, AXIS y MODULE.

También debe especificar la cantidad de números enteros de 16 bits (INT) que desea enviar o recibir. Puede especificar desde 0 hasta 65 números enteros. Si especifica 0 para un mensaje BTR, el módulo de transferencia en bloques determina cuántos números enteros de 16 bits se enviarán. Si especifica 0 para un mensaje BTW, el controlador envía 64 números enteros.

El módulo de E/S que recibirá la transferencia en bloques se debe identificar en el organizador de controlador.

Cuando se selecciona un tipo de mensaje de transferencia en bloques, no se selecciona un método de comunicación en la ficha Configuración. La selecciones CIP y DH+ aparecen en gris.

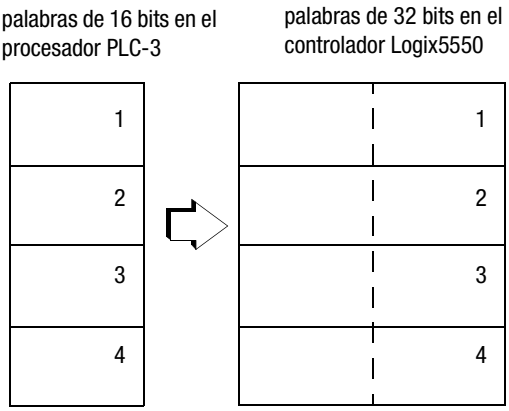
Cómo especificar los mensajes PLC-3

Los tipos de mensaje PLC-3 han sido diseñados para los procesadores PLC-3.

Seleccione este comando:	Si usted desea:
TypedRead PLC3	leer números enteros o datos de tipo REAL. Para los números enteros, este comando lee los números enteros de 16 bits del procesador PLC-3 (tipos de archivo S, B y N) y los almacena en arreglos de datos SINT, INT o DINT en el controlador Logix5550 y mantiene la integridad de los datos. Este comando también lee datos de punto flotante (coma flotante) del procesador PLC-3 (tipo de archivo F) y los almacena en un tag de tipo de datos REAL en el controlador Logix5550.
Typed Write PLC3	escribir números enteros o datos de tipo REAL. Este comando escribe los datos SINT o INT al archivo de enteros PLC-3 y mantiene la integridad de los datos. Puede escribir los datos DINT siempre que quepan dentro de un tipo de datos INT (–32,768 ≥ datos ≤ 32,767). Este comando también escribe los datos de tipo REAL del controlador Logix5550 hacia un archivo de punto flotante (coma flotante) PLC-3 (tipo de archivo F).
Lectura de rango de palabra PLC3	leer un rango contiguo de palabras de 16 bits en la memoria PLC-3 independientemente del tipo de datos. Este comando comienza a partir de la dirección especificada como el Elemento de origen y lee secuencialmente el número de palabras de 16 bits solicitadas. Los datos del Elemento de origen se almacenan a partir de la dirección especificada como el Tag de destino.
Escritura de rango de palabra PLC3	escribir un rango contiguo de palabras de 16 bits de la memoria Logix5550 independientemente del tipo de datos hacia le memoria PLC-3. Este comando comienza a partir de la dirección especificada como el tag de origen y lee secuencialmente el número de palabras de 16 bits solicitadas. Los datos del tag de origen se almacena a partir de la dirección especificada como el Elemento de origen en el procesador PLC-3.

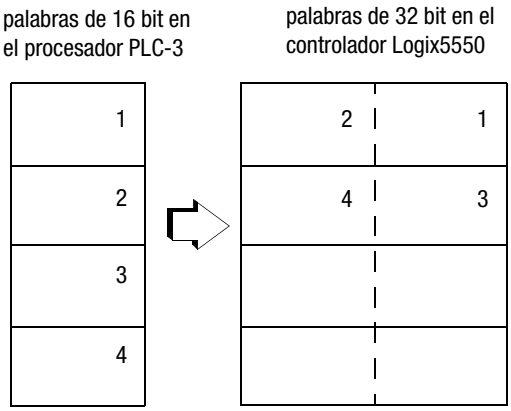
Los diagramas siguientes muestran la diferencias entre los comandos TypedWrite/TypedRead y de rango de palabra. El ejemplo usa los comandos de lectura de un procesador PLC-3 hacia un controlador Logix5550.

Comando TypedRead



Los comandos TypedWrite/TypedRead mantienen la estructura y valor de datos.

Comando de rango de palabra



Los comandos de rango de palabra llenan el tag de destino contiguamente. La estructura y valor de datos se cambian según el tipo de datos de destino.

Cómo especificar los mensajes PLC-2

Los tipos de mensaje PLC-2 han sido diseñados para los procesadores PLC-2.

Seleccione este comando:	Si usted desea:
Lectura no protegida PLC2	leer las palabras de 16 bits de cualquier zona desde la tabla de datos PLC-2 ó el archivo de compatibilidad PLC-2 de otro procesador.
Escritura no protegida PLC2	escribir las palabras de 16 bits a cualquier zona de la tabla de datos PLC-2 ó el archivo de compatibilidad PLC-2 de otro procesador.

La transferencia de mensajes usa palabras de 16 bits, por lo tanto, asegúrese de que el tag Logix5550 almacena apropiadamente los datos transferidos (típicamente como un arreglo INT).

Ejemplos de configuración MSG

Los ejemplos siguientes muestran ejemplos de tags de origen y destino además de elemento para diferentes combinaciones de controladores.

Para las instrucciones MSG que provienen de un controlador Logix5550 y escriben a otro procesador:

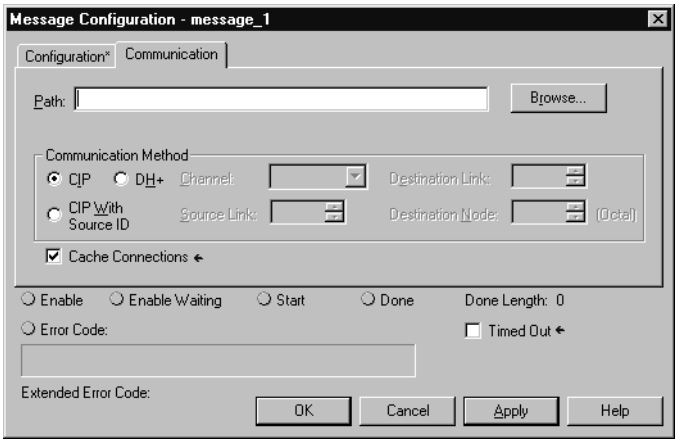
Ruta de mensaje:	Ejemplo de origen y destino:	
Logix5550 → Logix5550	tag de origen	*array_1*
	tag de destino	*array_2*
	Puede usar un tag de alias para el tag de origen (en el controlador Logix5550 de origen). Si se desea iniciar un offset dentro de un arreglo, hay que usar un alias para indicar el offset. **No puede** usar un alias para el tag de destino. El destino debe ser un tag base.	
Logix5550 → PLC-5 Logix5550 → SLC	tag de origen	*array_1*
	elemento de destino	*N7:10*
	Puede usar un tag de alias para el tag de origen (en el controlador Logix5550 de origen). Si se desea iniciar un offset dentro de un arreglo, hay que usar un alias para indicar el offset.	
Logix5550 → PLC-2	tag de origen	*array_1*
	elemento de destino	*010*

Para las instrucciones MSG que provienen de un controlador Logix5550 y leen de otro procesador:

Ruta de mensaje:	Ejemplo de origen y destino:	
Logix5550 → Logix5550	tag de origen	*array_1*
	tag de destino	*array_2*
	No puede usar un tag de alias para el tag de origen. El origen debe ser un tag base. Puede usar un tag de alias para el tag de destino (en el controlador Logix5550 de origen). Si se desea iniciar un offset dentro de un arreglo, hay que usar un alias para indicar el offset.	
Logix5550 → PLC-5 Logix5550 → SLC	elemento de origen	*N7:10*
	tag de destino	*array_1*
	Puede usar un tag de alias para el tag de destino (en el controlador Logix5550 de origen). Si se desea iniciar un offset dentro de un arreglo, hay que usar un alias para indicar el offset.	
Logix5550 → PLC-2	elemento de origen	*010*
	tag de destino	*array_1*

Cómo especificar los detalles de comunicación (ficha Comunicación)

Cuando usted configura una instrucción MSG, configura estos detalles en la ficha Comunicación.

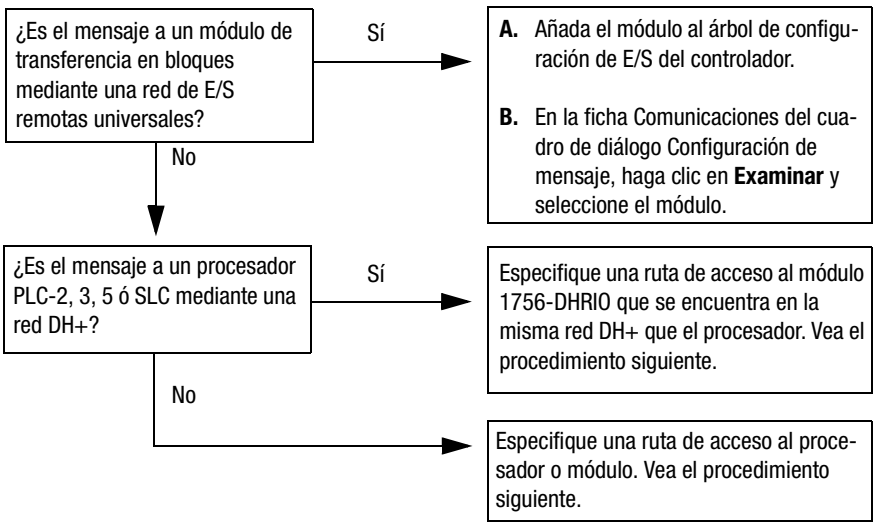


Cómo especificar una ruta de conexión

La ruta de conexión describe la ruta que sigue el mensaje para llegar al procesador o módulo de destino. La ruta usa números, donde parejas de números llegan al usuario de un módulo a otro mediante un backplane ControlLogix o una red de comunicaciones.

▶ Si el árbol de configuración de E/S contiene el procesador o módulo de destino en lugar de especificar una ruta, haga clic en **Examinar** para seleccionar el dispositivo.

Use el siguiente diagrama de flujo para determinar la necesidad de introducir una ruta.


¿Es el mensaje a un módulo de transferencia en bloques mediante una red de E/S remotas universales?
Sí
A. Añada el módulo al árbol de configuración de E/S del controlador.
B. En la ficha Comunicaciones del cuadro de diálogo Configuración de mensaje, haga clic en Examinar y seleccione el módulo.
No
¿Es el mensaje a un procesador PLC-2, 3, 5 ó SLC mediante una red DH+?
Sí
Especifique una ruta de acceso al módulo 1756-DHRIO que se encuentra en la misma red DH+ que el procesador. Vea el procedimiento siguiente.
No
Especifique una ruta de acceso al procesador o módulo. Vea el procedimiento siguiente.

Para especificar una ruta de acceso:

1. En el cuadro de texto Ruta de acceso, escriba lo siguiente, separando cada número con una coma [,]:

A. Especifique el puerto por el cual sale el mensaje. (Para el primer número en la ruta, esto es generalmente "1", el cual asigna el puerto del backplane del controlador Logix5550 que está enviando el mensaje.):

Para el:	Especifique:
backplane de cualquier procesador o módulo 1756	1
puerto DF1 desde un controlador 1756-L1	2
puerto ControlNet de un módulo 1756-CNB	
puerto Ethernet desde un módulo 1756-ENET	
puerto DH+ mediante canal A desde un módulo 1756-DHRIO	
puerto DH+ mediante canal B desde un módulo 1756-DHRIO	3

B. Especifique el módulo:

Para un módulo en:	Especifique:
backplane ControlBus	número de ranura
red DF1	dirección de estación (0 – 254)
red ControlNet	número de nodo (1 – 99 decimal)
red DH+	número de nodo (1 – 77 octal)
red Ethernet	dirección IP (cuatro números decimales separados por puntos)

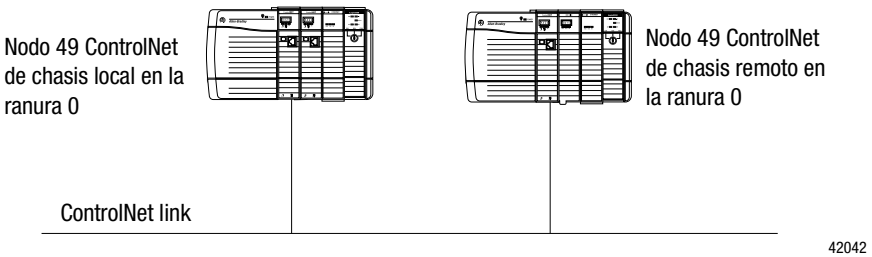
2. Repita el paso 1. hasta especificar el procesador o módulo de destino.

Las páginas siguientes presentan ejemplos de rutas de conexión:

- mediante ControlNet, vea la página 3-23
- mediante Ethernet, vea la página 3-24
- para un mensaje DH+, vea la página 3-24

Los dos ejemplos siguientes muestran las rutas de conexión entre los controladores Logix5550 mediante las redes ControlNet y Ethernet. El controlador que envía el mensaje se encuentra en el chasis local y el controlador que recibe el mensaje se encuentra en el chasis remoto.

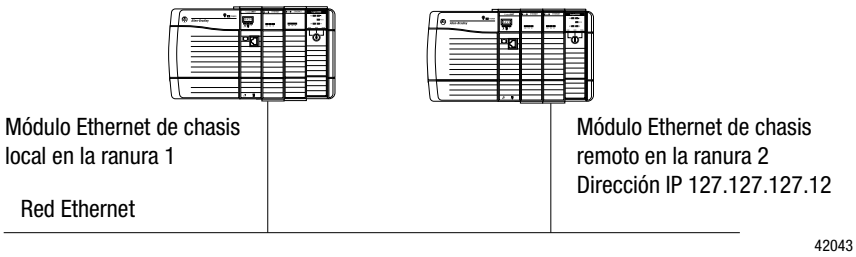
Ejemplo 1: Cómo especificar una ruta de acceso mediante ControlNet


Nodo 49 ControlNet de chasis local en la ranura 0
Nodo 49 ControlNet de chasis remoto en la ranura 0
ControlNet link
42042

Ruta de acceso: 1, 0, 2, 42, 1, 3

donde:	**Indica:**
1	el puerto del backplane del controlador Logix5550 en el chasis local
0	el número de ranura del módulo 1756-CNB en el chasis local
2	el puerto ControlNet del módulo 1756-CNB en ranura 0 del chasis local
42	el nodo ControlNet del módulo 1756-CNB en ranura 0 del chasis remoto
1	el puerto del backplane del módulo 1756-CNB en el chasis remoto
3	el número de ranura el controlador en el chasis remoto

Ejemplo 2: Cómo especificar una ruta de acceso mediante Ethernet

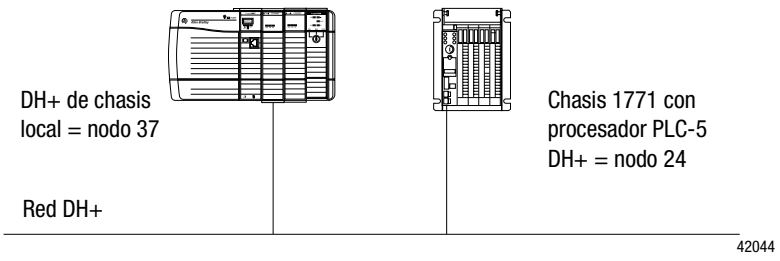

Módulo Ethernet de chasis local en la ranura 1
Red Ethernet
Módulo Ethernet de chasis remoto en la ranura 2
Dirección IP 127.127.127.12
42043

Ruta de acceso: 1, 1, 2, 127.127.127.12, 1, 3

donde:	**Indica:**
1	el puerto del backplane del controlador Logix5550 en el chasis local
1	el número de ranura del módulo 1756-ENET en el chasis local
2	el puerto Ethernet del módulo 1756-ENET en el chasis local
127.127.127.12	la dirección IP del módulo 1756-ENET en el chasis remoto
1	el puerto del backplane del módulo 1756-ENET en el chasis remoto
3	el número de ranura el controlador en el chasis remoto

El ejemplo siguiente muestra la ruta de conexión usada en un mensaje a un procesador PLC-5. En este caso, la ruta se termina con el módulo 1756-DHRIO que se encuentra en la misma red DH+ que el procesador PLC-5.

Ejemplo 3: Cómo especificar una ruta de acceso para un mensaje DH+



Ruta de acceso: 1, 1

donde:	**Indica:**
1	el puerto del backplane del controlador Logix5550 en el chasis local
1	el número de ranura del módulo 1756-DHRIO en el chasis local

Cómo especificar un método de comunicación:

Use la tabla siguiente para seleccionar un método de comunicación para el mensaje.

▶ Para un tipo de mensaje CIP genérico o de transferencia en bloques, *no* es necesario seleccionar un método de comunicación. (La sección no está disponible).

Si el dispositivo de destino es un:	Entonces seleccione:	Y especifique:	
controlador ControlLogix	**protocolo de control e información**	no se requieren otras especificaciones	
procesador ControlNet PLC-5			
Procesador PLC-5	**DH+**	**Canal:**	El canal A o B del módulo 1756-DHRIO que está conectado a la red DH+
procesador SLC		**Vínculo de origen:**	La identificación de vínculo asignada al backplane del controlador en la tabla de encaminamiento del módulo 1756-DHRIO (El nodo de origen en la tabla de encaminamiento es automáticamente el número de ranura del controlador.)
Procesador PLC-3		**Vínculo de destino**	La identificación de vínculo de la red DH+ remota donde reside el dispositivo receptor
Procesador PLC-2		**Nodo de destino:**	La dirección de estación del dispositivo receptor
		Si hay solamente una red DH+ y no se usa el software Gateway para configurar el módulo DH/RIO para vínculos remotos, hay que especificar 0 para el vínculo de origen y el vínculo de destino.	
Aplicación en una estación de trabajo que recibe un mensaje no solicitado encaminado mediante una red Ethernet o ControlNet a través de RSLinx.	**CIP con ID de origen** (Esto permite que la aplicación reciba datos de un controlador.)	**Vínculo de origen:**	La identificación de vínculo de los datos (0 – 65535) proporcionados por la aplicación a RSLinx
		Vínculo de destino:	La identificación de vínculo virtual configurado en RSLinx (0 – 65535)
		Nodo de destino:	La identificación de destino (0 – 77 octal) proporcionada por la aplicación a RSLinx
		El número de ranura del controlador Logix5550 se usa como el Nodo de origen.	

Cómo seleccionar una opción de caché:

Algunos timpos de mensajes usan una conexión para enviar o recibir datos.Hay otros que también ofrecen la opción de dejar abierta la conexión (caché) o cerrar la conexión una vez transmitido el mensaje. La tabla siguiente indica qué mensajes usan una conexión y si es posible almacenar en caché la conexión:

Este tipo de mensaje:	Usando este método de comunicación:	Usa una conexión:	Que se puede almacenar en caché:
lectura o escritura de la tabla de datos CIP	protocolo de control e información	✓	✓
PLC2, PLC3, PLC5 ó SLC (todos los tipos)	protocolo de control e información		
	CIP con ID de origen		
	DH+	✓	
CIP genérico	N/A		
transferencia en bloques de lectura o escritura	N/A	✓	✓

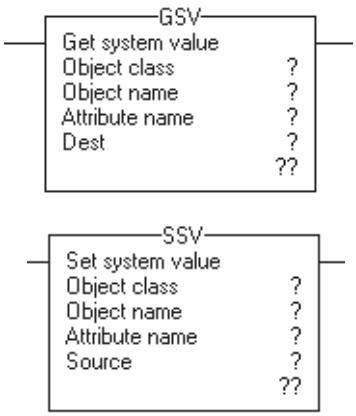
Use la tabla siguiente para seleccionar una opción de caché para un mensaje.

Si el mensaje se ejecuta:	Entonces:	Porque:
repetidamente	Seleccione la casilla de verificación Conexiones de caché	Esto mantiene abierta la conexión y optimiza el tiempo de ejecución. El abrir una conexión cada vez que se ejecuta el mensaje aumenta el tiempo de ejecución.
con poca frecuencia	Borre la casilla de verificación Conexiones de caché	Esto cerrará la conexión una vez completada, lo cual hace la conexión disponible para otros usos.

Get System Value (GSV) y Set System Value (SSV)

La instrucción GSV es una instrucción de salida.
La instrucción GSV es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Instrucción GSV			
Clase de objeto		nombre	nombre de la clase de objeto
Nombre del objeto		nombre	nombre de objeto específico cuando el objeto requiere un nombre
Nombre de atributo		nombre	atributo del objeto el tipo de datos depende del atributo que usted selecciona
Destino	SINT INT DINT REAL	tag	destino para los datos del atributo
Instrucción SSV			
Clase de objeto		nombre	nombre de la clase de objeto
Nombre del objeto		nombre	nombre de objeto específico cuando el objeto requiere un nombre
Nombre de atributo		nombre	atributo del objeto
Origen	SINT INT DINT REAL	tag	el tag que contiene los datos que desea copiar al atributo

Descripción: Las instrucciones GSV/SSV reciben y envían datos de sistema del controlador que se almacenan en los objetos. El controlador almacena datos de sistema en los objetos. No hay un archivo de estado, a diferencia del procesador PLC-5.

Cuando está habilitada, la instrucción GSV recupera la información especificada y la coloca en el destino. Cuando está habilitada, la instrucción SSV establece el atributo especificado con datos del origen.

Cuando se introduce una instrucción GSV/SSV, el software de programación muestra en pantalla las clases de objetos, nombres de objetos y nombres de atributos válidos para cada instrucción. En cuanto a la instrucción GSV, es posible obtener los valores para todos los atributos disponibles. Para la instrucción SSV, el software muestra en pantalla solamente los atributos que se permiten establecer (SSV).

ATENCION: use la instrucción GSV/SSV con cuidado. El hacer cambios a los objetos puede causar una operación inesperada del controlador o lesiones personales.

Si el tamaño del Origen o Destino es demasiado pequeño, la instrucción no se ejecuta y se registra un fallo menor. La sección siguiente, *Objetos GSV/SSV*, define los atributos de cada objeto y los tipos de datos asociados. Por ejemplo, el atributo MajorFaultRecord del objeto Program requiere un tipo de datos DINT[11].

Ejecución:

Condición:	Acción:
preescán	La condición de renglón de salida está establecida como falsa.
la condición de renglón precedente es falsa	La condición de renglón de salida está establecida como falsa.
la condición de renglón precedente es verdadera	Obtener o definir el valor especificado. La condición de renglón de salida está establecida como verdadera

Indicadores de estado aritmético: no afectado

Condiciones de fallo:

Ocurrirá un fallo menor si:	Tipo de fallo:	Código de fallo:
dirección de objeto no válida	4	5
se especificó un objeto que no es compatible con GSV/SSV	4	6
atributo no válido	4	6
no se proporcionó suficiente información para una instrucción SSV	4	6
el destino GSV no es suficientemente grande para retener los datos solicitados	4	7

Otros formatos:

Formato:	Sintaxis:
texto neutro	`GSV(object_class,object_name,attribute_name,destination);` `SSV(object_class,object_name,attribute_name,destination);`
texto ASCII	`GSV object_class object name attribute_name destination` `SSV object_class object name attribute_name destination`

Objetos GSV/SSV

Cuando se introduce una instrucción GSV/SSV, se especifica el objeto, y el atributo del mismo, al cual se desea obtener acceso. En ciertos casos, existirá más de una instancia del mismo tipo de objeto, por lo tanto, también puede ser necesario especificar el nombre del objeto. Por ejemplo, la aplicación puede tener diversas tareas. Cada tarea tiene su propio objeto TASK al cual se puede obtener acceso mediante el nombre de la tarea.

ATENCION: Para la instrucción GSV, se copian al destino solamente el tamaño especificado de datos. Por ejemplo, si el atributo especificado es SINT y el destino es DINT, se actualizan solamente los 8 bits inferiores del destino DINT y los otros 24 bits no se cambian.

Se puede obtener acceso a los objetos siguientes:

Para obtener información acerca de este objeto:	Vea la página:
AXIS	3-30
CONTROLLER	3-36
CONTROLLERDEVICE	3-36
CST	3-38
DF1	3-39
FAULTLOG	3-42
MESSAGE	3-42
MODULE	3-44
MOTIONGROUP	3-45
PROGRAM	3-46
ROUTINE	3-46
SERIALPORT	3-47
TASK	3-48
WALLCLOCKTIME	3-49

Cómo obtener acceso al objeto AXIS

El objeto AXIS proporciona información de estado acerca de un eje de módulo de servo. Especifique el nombre del tag de eje para seleccionar el objeto AXIS deseado.

Vea la publicación 1756-6.5.16, *Logix5550 Motion Module User Manual*, para obtener más información acerca del objeto AXIS.

Cuando un atributo está marcado con asterisco (*), significa que los atributos se encuentran en el controlador ControlLogix y el módulo de movimiento. Cuando usted usa una instrucción SSV para escribir uno de estos valores, el controlador actualiza automáticamente la copia en el módulo. Sin embargo, este proceso no se realiza inmediatamente. Para asegurarse de que el nuevo valor se actualiza en el módulo, use un mecanismo de enclavamiento usando bits booleanos en el UpdateStatus del tag Axis.

Por ejemplo, si realiza un comando SSV a PositionLockTolerance, el PositionToStatus del tag Axis se establece hasta que se actualiza con éxito el módulo. Por lo tanto, la lógica que sigue al comando SSV no puede esperar el restablecimiento de este bit antes de continuar en el programa.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
*AccelerationFeedforwardGain	REAL	GSV SSV	El valor usado para proporcionar la salida de comando de par a fin de generar la aceleración de comando.
ActualPosition	REAL	GSV	El posición real del eje.
ActualVelocity	REAL	GSV	La velocidad real del eje.
AverageVelocity	REAL	GSV	La velocidad promedio del eje.
AverageVelocityTimebase	REAL	GSV SSV	La base de tiempo de la velocidad promedio del eje.
AxisConfigurationState	SINT	GSV	El estado de configuración del eje.
AxisState	SINT	GSV	El estado operativo del eje. **Valor: Significado:** 0 eje listo 1 control directo de variador 2 control servo 3 fallo de eje 4 desactivación de eje
*AxisType	INT	GSV SSV	El tipo de eje que se usa. **Valor: Significado:** 0 eje no usado 1 eje de sólo posición 2 servoeje
C2CConnectionInstance	DINT	GSV	La ocurrencia de conexión del controlador que produce los datos de eje.
C2CMapTableInstance	DINT	GSV	La ocurrencia de asignación del controlador que produce los datos de eje.
CommandPosition	REAL	GSV	La posición de comando del eje.
CommandVelocity	REAL	GSV	La velocidad de comando del eje.
ConversionConstant	REAL	GSV SSV	El factor de conversión usado para realizar una conversión de las unidades en conteos de retroalimentación.
DampingFactor	REAL	GSV SSV	El valor usado para calcular el ancho de banda máximo de la posición servo durante la ejecución de la instrucción Motion Run Axis Tuning (MRAT).

Atributo:	Tipo de datos:	Instrucción:	Descripción:
*DriveFaultAction	SINT	GSV SSV	La operación que se realiza cuando ocurre un fallo del variador. **Valor:** **Significado:** 0 desactive el eje 1 inhabilite el variador 2 pare el movimiento ordenado 3 cambie el bit de estado solamente
EffectiveInertia	REAL	GSV	El valor de inercia para el eje calculado según las mediciones efectuadas por el controlador durante la última instrucción Motion Run Axis Tuning (MRAT).
*EncoderLossFaultAction	SINT	GSV SSV	La operación que se realiza cuando ocurre un fallo de pérdida de encoder. **Valor:** **Significado:** 0 desactive el eje 1 inhabilite el variador 2 pare el movimiento ordenado 3 cambie el bit de estado solamente
*EncoderNoiseFaultAction	SINT	GSV SSV	La operación que se realiza cuando ocurre un fallo de ruido de encoder. **Valor:** **Significado:** 0 desactive el eje 1 inhabilite el variador 2 pare el movimiento ordenado 3 cambie el bit de estado solamente
*FrictionCompensation	REAL	GSV SSV	El nivel de salida fijo usado para compensar la fricción estática.
GroupInstance	DINT	GSV	El número de instancia del grupo de movimiento que contiene el eje.
HomeMode	SINT	GSV SSV	El modo vuelta a la posición inicial del eje. **Valor:** **Significado:** 0 vuelta a la posición pasiva 1 vuelta a la posición activa (opción predeterminada)
HomePosition	REAL	GSV SSV	La posición inicial del eje.
HomeReturnSpeed	REAL	GSV SSV	La velocidad de vuelta a la posición inicial del eje.
HomeSequenceType	SINT	GSV SSV	El tipo de secuencia de vuelta a la posición inicial del eje. **Valor:** **Significado:** 0 vuelta a la posición inicial inmediata 1 vuelta a la posición inicial de interruptor 2 vuelta a la posición inicial de marcador 3 vuelta a al posición inicial de interruptor-marcador (predeterminada)
HomeSpeed	REAL	GSV SSV	La velocidad de vuelta a la posición inicial del eje.
Instance	DINT	GSV	El número de instancia del eje.
MapTableInstance	DINT	GSV	La instancia del mapa de E/S del módulo servo.
MaximumAcceleration	REAL	GSV SSV	La aceleración máxima del eje.
MaximumDeceleration	REAL	GSV SSV	La desaceleración máxima del eje.
*MaximumNegativeTravel	REAL	GSV SSV	El fin de carrera negativo máximo.
*MaximumPositiveTravel	REAL	GSV SSV	El fin de carrera positivo máximo.
MaximumSpeed	REAL	GSV SSV	La velocidad máxima del eje.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
ModuleChannel	SINT	GSV	El canal de módulo del módulo servo.
MotionConfigurationBits	DINT	GSV SSV	Los bits de configuración de movimiento para el eje. **bit:** **Significado:** 0 retroceso de dirección de vuelta a la posición inicial 1 interruptor de vuelta a la posición inicial normalmente cerrado 2 flanco marcado negativo de vuelta al posición inicial
MotionFaultBits	DINT	GSV	Los bits de fallo de movimiento para el eje. (En la estructura AXIS, éste es el miembro MotionFault.) **bit:** **Nombre del bit:** **Significado:** 0 ACAsyncConnFault fallo de conexión asíncrona 1 ACSyncConnFault fallo de conexión síncrona
MotionStatusBits	DINT	GSV	Los bits de estado de movimiento para el eje. (En la estructura AXIS, éste es el miembro MotionStatus.) **bit:** **Nombre del bit:** **Significado:** 0 AccelStatus aceleración 1 DecelStatus desaceleración 2 MoveStatus mover 3 JogStatus impulsar 4 GearingStatus transmisión 5 HomingStatus vuelta a la posición inicial 6 ClutchStatus embrague 7 AxisHomedStatus estado de vuelta a la posición inicial
MotorEncoderTestIncrement	REAL	GSV SSV	La cantidad de movimiento necesaria para iniciar la prueba Motion Run Hookup Diagnostic (MRHD).
*OutputFilterBandwidth	REAL	GSV SSV	El ancho de banda del filtro servo de salida digital de paso bajo.
*OutputLimit	REAL	GSV SSV	El valor del voltaje máximo de salida servo del eje.
*OutputOffset	REAL	GSV SSV	El valor usado para compensar los efectos de los offsets acumulativos de la salida DAC del módulo servo y la entrada del servovariador.
*OutputScaling	REAL	GSV SSV	El valor usado para convertir la salida del lazo servo en el voltaje equivalente al variador.
PositionError	REAl	GSV	La diferencia entre la posición real y la posición de comando de un eje.
*PositionErrorFaultAction	SINT	GSV SSV	La operación que se realiza cuando ocurre un fallo de error de posición. **Valor:** **Significado:** 0 desactive el eje 1 inhabilite el variador 2 pare el movimiento ordenado 3 cambie el bit de estado solamente
*PositionErrorTolerance	REAL	GSV SSV	La cantidad de errores de posición que el servo tolera antes de emitir un fallo de error de posición.
*PositionIntegralGain	REAL	GSV SSV	El valor usado para lograr el posicionamiento de eje preciso a pesar de las perturbaciones, tales como la fricción estática y la gravedad.
PositionIntegratorError	REAL	GSV	La suma del error de posición para un eje.
PositionLockTolerance	REAL	GSV SSV	La cantidad de errores de posición que el módulo servo tolera cuando indica un estado de posición verdadera bloqueada.
*PositionProportionalGain	REAL	GSV SSV	El valor que el controlador multiplica con el error de posición para compensar el error de posición.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
PositionServoBandwidth	REAL	GSV SSV	El ancho de banda de ganancia de unidad que el controlador usa para calcular las ganancias para una instrucción Motion Apply Axis Tuning (MAAT).
*PositionUnwind	DINT	GSV SSV	El valor usado para realizar el desbobinado automático del eje rotativo.
ProgrammedStopMode	SINT	GSV SSV	El tipo de detención que se realiza en el eje. **Valor:** **Significado:** 0 detención rápida 1 desactivación rápida 2 desactivación inmediata
RegistrationPosition	REAL	GSV	La posición de registro para el eje.
*ServoConfigurationBits	DINT	GSV SSV	Los bits de configuración servo para el lazo del servo. **bit:** **Significado:** 0 eje rotativo 1 servovariador de velocidad externa 2 polaridad negativa de encoder 3 polaridad servo negativa 4 verificación de sobrecarrera basada en software 5 verificación de error de posición 6 verificación de fallo de pérdida de encoder 7 verificación de fallo de ruido de encoder 8 verificación de fallo del variador 9 fallo del variador normalmente cerrado
ServoConfigurationUpdateBits	DINT	GSV	Los bits de estado de configuración servo para el lazo del servo. (En la estructura AXIS, éste es el miembro UpdateStatus.) **bit:** **Nombre del bit:** **Significado:** 0 AxisTypeStatus tipo de eje 1 PosUnwndStatus posición de desbobinado 2 MaxPTrvlStatus carrera positiva máxima 3 MaxNTrvlStatus carrera negativa máxima 4 PosErrorTolStatus tolerancia de error de posición 5 PosLockTolStatus tolerancia de bloqueo de posición 6 PosPGainStatus ganancia proporcional de posición 7 PosIGainStatus ganancia integral de posición 8 VelFfGainStatus ganancia de prealimentación de velocidad 9 AccFfGainStatus prealimentación de aceleración ganancia 10 VelPGainStatus ganancia proporcional de velocidad 11 VelIGainStatus ganancia integral de velocidad 12 OutFiltBwStatus ancho de banda de filtro de salida 13 OutScaleStatus escalado de salida 14 OutLimitStatus límite de salida 15 OutOffsetStatus offset de salida 16 FricCompStatus compensación de fricción 17 POtrvlFaultActStatus acción de fallo de sobrecarrera basada en software 18 PosErrorFaultActStatus acción de fallo de error de posición 19 EncLossFaultActStatus acción de fallo de pérdida de encoder 20 EncNsFaultActStatus acción de fallo de ruido de encoder 21 DriveFaultActStatus acción de fallo del variador

Atributo:	Tipo de datos:	Instrucción:	Descripción:
ServoEventBits	DINT	GSV	Los bits de evento servo para el lazo del servo. (En la estructura AXIS, éste es el miembro EventStatus.) **bit:** **Nombre del bit:** **Significado:** 0 WatchEvArmStatus evento de vigilancia activado 1 WatchEvStatus evento de vigilancia 2 RegEvArmStatus evento de registro activado 3 RegEvStatus evento de registro 4 HomeEvArmStatus evento de vuelta a la posición inicial activado 5 HomeEvStatus evento de vuelta a la posición inicial
ServoFaultBits	DINT	GSV	Los bits de fallo servo para el lazo del servo. (En la estructura AXIS, éste es el miembro ServoFault.) **bit:** **Nombre del bit:** **Significado:** 0 POtrvlFault fallo de sobrecarrera positivo 1 NOtrvlFault fallo de sobrecarrera negativo 2 PosErrorFault fallo de error de posición 3 EncCHALossFault fallo de pérdida de canal A del encoder 4 EncCHBLossFault fallo de pérdida de canal B del encoder 5 EncCHZLossFault fallo de pérdida de canal Z del encoder 6 EncNsFault fallo de ruido de encoder 7 DriveFault fallo del variador 8 SyncConnFault fallo de conexión síncrono 9 HardFault fallo de hardware servo
ServoOutputLevel	REA;	GSV	El nivel de voltaje de salida para el lazo del servo de eje.
ServoStatusBits	DINT	GSV	Los bits de estado para el lazo del servo. (En la estructura AXIS, éste es el miembro ServoStatus.) **bit:** **Nombre del bit:** **Significado:** 0 ServoActStatus acción del servo 1 DriveEnableStatus habilitación del variador 2 OutLmtStatus límite de salida 3 PosLockStatus bloqueo de posición 13 TuneStatus proceso de ajuste 14 TestStatus diagnóstico de prueba 15 ShutdownStatus desactivación del eje
ServoStatusUpdateBits	DINT	GSV	Los bits de actualización de estado del servo para el eje. **bit:** **Significado:** 0 actualización del error de posición 1 actualización del error de integrador de posición 2 actualización del error de velocidad 3 actualización del error de integrador de velocidad 4 actualización de comando de velocidad 5 actualización de retroalimentación de velocidad 6 actualización de nivel de salida del servo
*SoftOvertravelFaultAction	SINT	GSV SSV	La operación que se realiza cuando ocurre un fallo sobrecarrera basada en software. **Valor:** **Significado:** 0 desactive el eje 1 inhabilite el variador 2 pare el movimiento ordenado 3 cambie el bit de estado solamente
StartActualPosition	REAL	GSV	La posición real del eje cuando empieza el nuevo movimiento ordenado para el eje.
StartCommandPosition	REAL	GSV	La posición de comando del eje cuando empieza el nuevo movimiento ordenado para el eje.
StrobeActualPosition	REAL	GSV	La posición real del eje cuando se ejecuta la instrucción Motion Group Strobe Position (MGSP).

Atributo:	Tipo de datos:	Instrucción:	Descripción:
StrobeCommandPosition	REAL	GSV	La posición de comando del eje cuando se ejecuta la instrucción Motion Group Strobe Position (MGSP).
TestDirectionForward	SINT	GSV	La dirección de carrera del eje durante la instrucción Motion Run Hookup Diagnostic (MRHD) según el módulo de servo. **Valor: Significado:** 0 dirección negativa (retroceso) 1 (ff) dirección positiva (adelante)
TestStatus	INT	GSV	El estado de la última instrucción Motion Run Hookup Diagnostic (MRHD). **Valor: Significado:** 0 el proceso de prueba se realizó con éxito 1 prueba en progreso 2 proceso de prueba cancelado por el usuario 3 la prueba sobrepasó el tiempo de espera de 2 segundos 4 fallo del proceso de prueba debido a un fallo del servo 5 incremento de prueba insuficiente
TuneAcceleration	REAL	GSV	El valor de aceleración medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneAccelerationTime	REAL	GSV	El tiempo de aceleración medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneDeceleration	REAL	GSV	El valor de desaceleración medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneDecelerationTime	REAL	GSV	El tiempo de desaceleración medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneRiseTime	REAL	GSV	El tiempo de elevación del eje medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneSpeedScaling	REAL	GSV	El factor de escalado de variador del eje medido durante la última instrucción Motion Run Axis Tuning (MRAT).
TuneStatus	INT	GSV	El estado de la última instrucción Axis Tuning (MRAT). **Valor: Significado:** 0 proceso de ajuste se realizó con éxito 1 ajuste en progreso 2 proceso de ajuste cancelado por el usuario 3 ajuste excedió el tiempo de espera de 2 segundos 4 proceso de ajuste entró en fallo debido a un fallo del servo 5 eje alcanzó el límite de fin de carrera 6 polaridad del eje establecida incorrectamente 7 velocidad de ajuste es demasiado pequeña para realizar mediciones
TuneVelocityBandwidth	REAL	GSV	El ancho de banda del variador calculado según las mediciones efectuadas por el controlador durante la última instrucción Motion Run Axis Tuning (MRAT).
TuningConfigurationBits	DINT	GSV SSV	Los bits de configuración de ajuste para el eje. **bit: Significado:** 0 dirección de ajuste (0 = avance, 1 = retroceso 1 integrador de error de posición de ajuste 2 integrador de error de velocidad de ajuste 3 bit de prealimentación de velocidad de ajuste 4 prealimentación de aceleración 5 filtro de paso bajo de velocidad de ajuste
TuningSpeed	REAL	GSV SSV	La velocidad máxima iniciada por la instrucción Motion Run Axis Tuning (MRAT).
TuningTravelLimit	REAL	GSV SSV	El fin de carrera usado por la instrucción Motion Run Axis Tuning (MRAT) para limitar la acción durante el ajuste.
VelocityCommand	REAL	GSV	La referencia de velocidad de corriente al lazo del servo de velocidad para un eje.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
VelocityError	REAL	GSV	La diferencia entre la velocidad ordenada y la velocidad real de un servoeje.
VelocityFeedback	REAL	GSV	La velocidad real del eje según la calcula el módulo servo.
*VelocityFeedforwardGain	REAL	GSV SSV	La salida de comando de velocidad necesaria para generar la velocidad ordenada.
*VelocityIntegralGain	REAL	GSV SSV	El valor que el controlador multiplica con el valor VelocityIntegratorError para corregir el error de velocidad.
VelocityIntegratorError	REAL	GSV	La suma del error de velocidad para un eje especificado.
*VelocityProportionalGain	REAL	GSV SSV	El valor que el controlador multiplica con el valor VelocityError para corregir el error de velocidad.
WatchPosition	REAL	GSV	El posición de control del eje.

Cómo obtener acceso al objeto CONTROLLER

El objeto CONTROLLER proporciona información de estado acerca la ejecución del controlador.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
TimeSlice	INT	GSV SSV	El porcentaje de la CPU disponible que se asigna a las comunicaciones. Los valores válidos son 10 – 90. No se puede cambiar este valor cuando el interruptor de llave del controlador está en la posición marcha.

Cómo obtener acceso al objeto CONTROLLERDEVICE

El objeto CONTROLLERDEVICE identifica el hardware físico del controlador.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
DeviceName	SINT[33]	GSV	Nombre de cadena ASCII del controlador. El primer byte contiene un conteo del número de los caracteres ASCII retornados en la cadena de arreglo.
ProductCode	INT	GSV	Identifica el tipo de controlador. Logix5550 = 3
ProductRev	INT	GSV	Identifica la revisión de producto actual. La visualización debe ser hexadecimal. El byte inferior contiene la revisión mayor; el byte superior contiene la revisión menor.
SerialNumber	DINT	GSV	Nombre de serie del dispositivo. Se asigna el número de serie cuando se construye el dispositivo.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Status	INT	GSV	Los bits identifican el estado: Los bits 3 – 0 están reservados **Bits de estado de dispositivo** **Bits 7 – 4: Significado:** 0000 reservado 0001 actualización de la memoria flash en progreso 0010 reservado 0011 reservado 0100 la memoria flash está defectuosa 0101 con fallo 0110 marcha 0111 programa **Bits de estado de fallo** **Bits 11 – 8: Significado:** 0001 fallo menor recuperable 0010 fallo menor no recuperable 0100 fallo mayor recuperable 1000 fallo mayor no recuperable **Bits de estado específicos para Logix5550** **Bits 13 – 12: Significado:** 01 interruptor de llave en marcha 10 interruptor de llave en programa 11 interruptor de llave en remoto **Bits 15 – 14: Significado:** 01 controlador cambia modos 10 depure el modo si el controlador está en el modo marcha
Type	INT	GSV	Identifica el dispositivo como controlador. Controlador = 14
Vendor	INT	GSV	Identifica el suministrador del dispositivo. Allen-Bradley = 0001

Cómo obtener acceso al objeto CST

El objeto CST (hora coordinada del sistema) proporciona la hora coordinada del sistema para los dispositivos en un chasis.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
CurrentStatus	INT	GSV	El estado actual de la hora coordinada del sistema. Los bits identifican:
CurrentValue	DINT[2]	GSV	El valor actual del temporizador. DINT[0] contiene los 32 bits inferiores; DINT[1] contiene los 32 bits superiores. El origen del temporizador se ajusta para coincidir con el valor suministrado en los servicios de actualización y la sincronización de red de comunicación local. El ajuste representa un aumento al valor solicitado o un establecimiento inmediato en el valor solicitado, según se indica en el atributo CurrentStatus.

bit:	Significado
0	**hardware del temporizador entró en fallo:** el hardware del temporizador interno del dispositivo está en un estado de fallo
1	**aumento habilitado:** el valor actual de los 16+ bits inferiores del temporizador aumentan al valor solicitado en lugar de ubicarse en el valor inferior. El método de sincronización por puntos para la red específica manipula estos bits.
2	**maestro de de la hora del sistema:** el objeto CST es el origen de la hora del maestro en el sistema ControlLogix
3	**sincronizado:** un objeto de CST maestro sincroniza el CurrentValue de 64 bits del objeto CST mediante una actualización de la hora del sistema
4	**maestro de red local:** el objeto CST es el origen de la hora del maestro de la red local
5	**en el modo relé:** el objeto CST funciona en un modo de relé de tiempo
6	**se ha detectado un maestro duplicado:** se ha detectado un maestro de hora duplicado de la red local. Este bit siempre es 0 para los nodos dependientes del tiempo.
7	no se usa
8 – 9	00 = nodo dependiente del tiempo 01 = nodo maestro de tiempo 10 = nodo de relé de tiempo 11 = no se usa
10 – 15	no se usa

Cómo obtener acceso al objeto DF1

El objeto DF1 proporciona una interface al controlador de comunicación que se puede configurar para el puerto en serie.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
ACKTimeout	DINT	GSV	La cantidad de tiempo necesaria para esperar una confirmación de una transmisión de mensaje (punto a punto y maestro solamente). El valor válido es 0 – 32,767. Retardo de conteos de períodos de 20 mseg. El valor predeterminado es 50 (1 segundo).
DiagnosticCounters	INT[19]	GSV	Arreglo de contadores diagnósticos para el variador de comunicación DF1.

offset de palabra	punto a punto DF1	esclavo DF1	maestro
0	firma (0 x 0043)	firma (0 x 0042)	firma (0 x 0044)
1	bits de módem	bits de módem	bits de módem
2	paquetes enviados	paquetes enviados	paquetes enviados
3	paquetes recibidos	paquetes recibidos	paquetes recibidos
4	paquetes no entregados	paquetes no entregados	paquetes no entregados
5	no se usa	reintentos de mensajes	reintentos de mensajes
6	NAK recibidos	NAK recibidos	no se usa
7	ENQ enviados	paquetes de encuesta recibidos	no se usa
8	paquetes deficientes NAK	paquetes deficientes sin ACK	paquetes deficientes sin ACK
9	memoria no enviada sin NAK	ninguna memoria sin ACK	no se usa
10	paquetes duplicados recibidos	paquetes duplicados recibidos	paquetes duplicados recibidos
11	caracteres deficientes recibidos	no se usa	no se usa
12	conteo de recuperaciones DCD	conteo de recuperaciones DCD	conteo de recuperaciones DCD
13	conteo de módem perdido	conteo de módem perdido	conteo de módem perdido
14	no se usa	no se usa	máximo de tiempo de escán de prioridad
15	no se usa	no se usa	último tiempo de escán de prioridad
16	no se usa	no se usa	máximo de tiempo de escán normal
17	no se usa	no se usa	último tiempo de escán normal
18	ENQ enviados	no se usa	no se usa

DuplicateDetection	SINT	GSV	Habilita la detección de mensajes duplicados. **Valor:** **Significado:** 0 detección de mensajes duplicados inhabilitada sin cero detección de mensajes duplicados inhabilitada
EmbeddedResponseEnable	SINT	GSV	Habilita la funcionalidad de respuesta incorporada (punto a punto solamente). **Valor:** **Significado:** 0 se inicia solamente después de que se recibe uno (como opción predeterminada) 1 habilitación sin condiciones
ENQTransmitLimit	SINT	GSV	El número de indagaciones (ENQ) que se envían después del tiempo de espera ACK (punto a punto solamente). El valor válido es 0 – 127. El establecimiento predeterminado es 3.
EOTSuppression	SINT	GSV	Habilita la supresión de transmisiones EOT como respuesta a los paquetes de encuesta (esclavo solamente). **Valor:** **Significado:** 0 supresión EOT inhabilitada sin cero supresión EOT habilitada
ErrorDetection	SINT	GSV	Especifica el esquema de detección de errores. **Valor:** **Significado:** 0 BCC (predeterminado) 1 CRC

Atributo:	Tipo de datos:	Instrucción:	Descripción:
MasterMessageTransmit	SINT	GSV	El valor actual de la transmisión del maestro de mensaje (maestro solamente). **Valor:** **Significado:** 0 entre encuestas de estación 1 en secuencia de encuesta (en lugar del número de estación del maestro) El valor predeterminado es 0.
NAKReceiveLimit	SINT	GSV	El número de NAK recibidos como respuesta a un mensajes antes de detener la transmisión (comunicación de punto a punto solamente). El valor válido es 0 – 127. El establecimiento predeterminado es 3.
NormalPollGroupSize	INT	GSV	El número de estaciones que se encuestan en el arreglo de nodos de encuesta normal después de encuestarse todas las estaciones en el arreglo de nodos de encuesta de prioridad (maestro solamente). El valor válido es 0 – 255. El establecimiento predeterminado es 0.
PollingMode	SINT	GSV	El modo de encuesta actual (maestro solamente). **Valor:** **Significado:** 0 basado en mensajes; no permite esclavos para iniciar mensajes 1 basado en mensajes; permite que los esclavos inicien mensajes (predeterminado) 2 transferencia de estándar de mensaje único por escán de nodo 3 transferencia de estándar de múltiples mensajes por escán de nodo El establecimiento predeterminado es 1.
ReplyMessageWait	DINT	GSV	El tiempo (actuando como maestro) que se debe esperar después de recibirse un ACK antes de encuestarse el esclavo en busca de una respuesta (maestro solamente). El valor válido es 0 – 65,535. Retardo de conteos de períodos de 20 mseg. El establecimiento predeterminado es 5 períodos (100 mseg).
StationAddress	INT	GSV	Dirección de estación actual del puerto en serie. El valor válido es 0 – 254. El establecimiento predeterminado es 0.
SlavePollTimeout	DINT	GSV	La cantidad de tiempo en mseg que espera el esclavo hasta que el maestro realiza una encuesta antes de que el esclavo indique que no puede transmitir debido a la inactividad del maestro (esclavo solamente). El valor válido es 0 – 32,767. Retardo de conteos de períodos de 20 mseg. El establecimiento predeterminado es 3000 períodos (1 minuto).
TransmitRetries	SINT	GSV	El número de veces que se puede volver a enviar un mensaje sin obtenerse una confirmación (maestro y esclavo solamente). El valor válido es 0 – 127. El establecimiento predeterminado es 3.
PendingACKTimeout	DINT	SSV	Valor pendiente para el atributo ACKTimeout
PendingDuplicateDetection	SINT	SSV	Valor pendiente para el atributo DuplicateDetection.
PendingEmbeddedResponse Enable	SINT	SSV	Valor pendiente para el atributo EmbeddedResponse.
PendingENQTransmitLimit	SINT	SSV	Valor pendiente para el atributo ENQTransmitLimit.
PendingEOTSuppression	SINT	SSV	Valor pendiente para el atributo EOTSuppression.
PendingErrorDetection	SINT	SSV	Valor pendiente para el atributo ErrorDetection.
PendingNormalPollGroupSize	INT	SSV	Valor pendiente para el atributo NormalPollGroupSize.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
PendingMasterMessage Transmit	SINT	SSV	Valor pendiente para el atributo MasterMessageTransmit.
PendingNAKReceiveLimit	SINT	SSV	Valor pendiente para el atributo NAKReceiveLimit.
PendingPollingMode	SINT	SSV	Valor pendiente para el atributo PollingMode.
PendingReplyMessageWait	DINT	SSV	Valor pendiente para el atributo ReplyMessageWait.
PendingStationAddress	INT	SSV	Valor pendiente para el atributo StationAddress.
PendingSlavePollTimeout	DINT	SSV	Valor pendiente para el atributo SlavePollTimeout.
PendingTransmitRetries	SINT	SSV	Valor pendiente para el atributo TransmitRetries.

Para aplicar los valores para cualquiera de los atributos DF1 pendientes:

1. Use una instrucción SSV para establecer el valor para el atributo pendiente.

   Usted puede establecer cualquier cantidad de atributos pendientes usando una instrucción SSV para cada atributo pendiente.

2. Use una instrucción MSG para aplicar el valor. La instrucción MSG aplica cada atributo pendiente establecido. Configure la instrucción MSG como:

Ficha Configuración MSG:	Campo:	Valor:
Configuración	Tipo de mensaje	CIP genérico
	Código de servicio	0d hex
	Tipo de objeto	a2
	ID del objeto	1
	Atributo del objeto	deje en blanco
	Origen	deje en blanco
	Número de elementos	0
	Destino	deje en blanco
Comunicación	Ruta de acceso	ruta de comunicación a sí misma (1, *s* donde *s* = número de ranura del controlador)

Cómo obtener acceso al objeto FAULTLOG

El objeto FAULTLOG proporciona información de fallo acerca del controlador.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
MajorEvents	INT	GSV SSV	Cuántos fallos mayores han ocurrido desde la última vez que se restableció este contador.
MinorEvents	INT	GSV SSV	Cuántos fallos menores han ocurrido desde la última vez que se restableció este contador.
MajorFaultBits	DINT	GSV SSV	Los bits individuales indican la razón del fallo mayor actual. **bit: Significado:** 1 corte de energía 3 E/S 4 ejecución de instrucción (programa) 5 administrador de fallos 6 temporizador de control (watchdog) 7 pila 8 cambio de modo 11 movimiento
MajorFaultBits	DINT	GSV SSV	Los bits individuales indican la razón del fallo menor actual. **bit: Significado:** 4 ejecución de instrucción (programa) 6 temporizador de control (watchdog) 9 puerto en serie 10 batería

Cómo obtener acceso al objeto MESSAGE

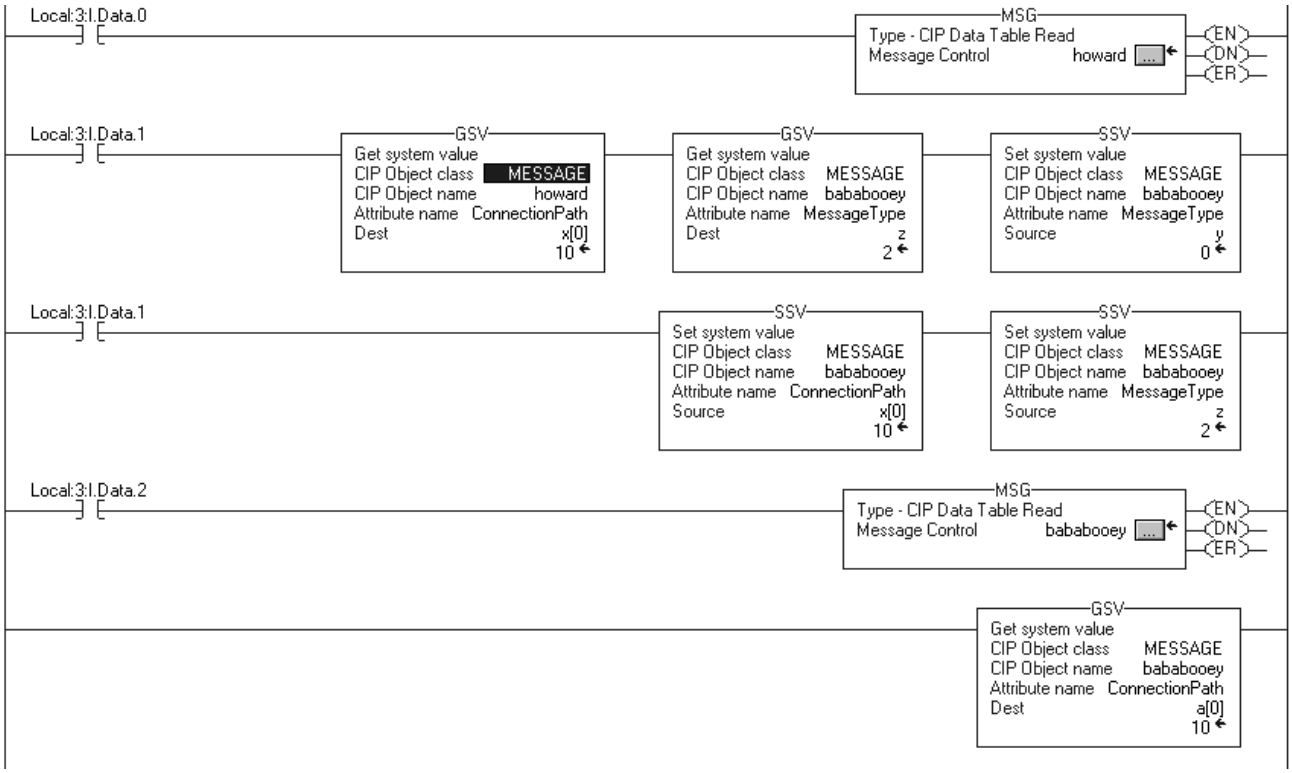
Usted puede obtener acceso al objeto MESSAGE mediante las instrucciones GSV/SSV. Especifique el nombre del tag de mensaje para determinar el objeto MESSAGE deseado. El objeto MESSAGE proporciona una interface para configurar y activar las comunicaciones entre dispositivos similares. Este objeto reemplaza el tipo de datos MSG del procesador PLC-5.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
ConnectionPath	SINT[130]	GSV SSV	Los datos necesarios para configurar la ruta de conexión. Los dos primeros bytes (byte inferior y byte superior) representan la longitud en bytes de la ruta de conexión.
ConnectionRate	DINT	GSV SSV	Régimen de paquete solicitado de la conexión.
MessageType	SINT	GSV SSV	Especifica el tipo de mensaje. **Valor: Significado:** 0 no inicializado
Puerto	SINT	GSV SSV	Indica el puerto en que se debe enviar el mensaje. **Valor: Significado:** 1 backplane 2 puerto en serie

Atributo:	Tipo de datos:	Instrucción:	Descripción:
TimeoutMultiplier	SINT	GSV SSV	Determina cuándo una conexión se debe considerar como que ha sobrepasado el tiempo de espera y cerrado. **Valor:** **Significado:** 0 la conexión sobrepasará el tiempo de espera en un lapso igual a 4 veces el régimen de actualización (predeterminado) 1 la conexión sobrepasará el tiempo de espera en un lapso igual a 8 veces el régimen de actualización 2 la conexión sobrepasará el tiempo de espera en un lapso igual a 16 veces el régimen de actualización
UnconnectedTimeout	DINT	GSV SSV	El tiempo de espera en microsegundos para todos los mensajes no conectados. La opción predeterminada es 30.000.000 microsegundos (30 segundos).

Siga los pasos a continuación para cambiar un atributo MESSAGE:

1. Use una instrucción GSV para obtener el atributo MessageType y guárdelo en un tag.
2. Use una instrucción SSV para establecer el MessageType en 0.
3. Use una instrucción SSV para establecer el atributo MESSAGE que desea cambiar.
4. Use una instrucción SSV para establecer el atributo MessageType nuevamente en el valor original que se obtuvo en el paso 1.



Cómo obtener acceso al objeto MODULE

El objeto MODULE proporciona información de estado acerca del módulo. Para seleccionar un objeto MODULE determinado, establezca el operando Nombre de objeto de la instrucción GSV/SSV en el nombre del módulo.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
EntryStatus	INT	GSV	Especifica el estado actual de la entrada de mapa especificada. Los 12 bits inferiores se deben enmascarar cuando se realiza una operación de comparación. Solamente los bits 12 – 15 son válidos. **Valor:** **Significado:** 16#0000 **De espera:** el controlador se enciende. 16#1000 **Con fallo:** cualquiera de las conexiones del objeto MODULE al módulo asociado entran en fallo. No se debe usar este valor para determinar si el módulo ha entrado en fallo puesto que el objeto MODULE sale periódicamente de este estado cuando intenta volver a conectarse al módulo. En cambio, realice una prueba para ver si hay un Estado de marcha (16#4000). Verifique si hay un FaultCode que no sea igual a 0 para determinar si un módulo ha entrado en fallo. Una vez entrados en fallo, los atributos FaultCode y FaultInfo son válidos hasta que se corrigen la condición de fallo. 16#2000 **Validación:** el objeto MODULE verifique la integridad del objeto MODULE antes de establecer las conexiones al módulo. 16#3000 **Conexión**: el objeto MODULE inicia las conexiones al módulo. 16#4000 **Se ejecuta:** todas las conexiones al módulo han sido establecidas y los datos se transfieren con éxito. 16#5000 **Desactivación:** el objeto MODULE está desactivando todas las conexiones al módulo. 16#6000 **Inhibido:** el objeto MODULE se inhibe (el bit de inhibición está establecido en el atributo de Modo). 16#7000 **De espera:** no funciona el objeto MODULE del cual depende este objeto MODULE.
FaultCode	INT	GSV	Un número que identifica un fallo del módulo, si ocurre.
FaultInfo	DINT	GSV	Especifica información específica acerca del código de fallo del objeto MODULE.
ForceStatus	INT	GSV	Especifica el estado de los forzados. **bit:** **Significado:** 0 forzados instalados (1 = sí, 0 = no) 1 forzados habilitados (1 = sí, 0 = no) 2 – 15 no se usa

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Instance	DINT	GSV	Proporciona el número de instancia de este objeto MODULE.
LEDStatus	INT	GSV	Especifica el estado actual del indicador LED de E/S en la parte frontal del controlador. **Valor:** **Significado:** 0 **Indicador LED apagado:** No hay objetos MODULE configurados para el controlador (no hay módulos en la sección Configuración de E/S del organizador de controlador). 1 Rojo parpadeante: No se ejecutan los objetos MODULE. 2 Verde parpadeante: No se ejecuta por lo menos un objeto MODULE. 3 Verde fijo: Se ejecutan todos los objetos del módulo. **Nota:** No introduzca un nombre de objeto con este atributo puesto que el atributo se aplica a la recolección toda de módulos.
Mode	INT	GSV SSV	Especifica el modo actual del objeto MODULE. **bit:** **Significado:** 0 Si está establecido, causa que se genere un fallo mayor si cualquiera de las conexiones del objeto MODULE entran en fallo mientras el controlador esté en el modo marcha. 2 Si está establecido, causa que el objeto MODULE entre en el estado inhibido después de desactivar todas las conexiones al módulo.

Cómo obtener acceso al objeto MOTIONGROUP

El objeto MOTIONGROUP proporciona información de estado acerca de un grupo de ejes para el módulo servo. Especifique el nombre de tag del grupo de movimiento para determinar el objeto MOTIONGROUP deseado.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Instance	DINT	GSV	Proporciona el número de instancia de este objeto MOTION_GROUP.

Cómo obtener acceso al objeto PROGRAM

El objeto PROGRAM proporciona información de estado acerca del programa. Especifique el nombre del programa de mensaje para determinar el objeto PROGRAM deseado.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
DisableFlag	SINT	GSV SSV	Controla la ejecución de este programa. **Valor:** **Significado:** 0 ejecución habilitada 1 ejecución inhabilitada
Instance	DINT	GSV	Proporciona el número de instancia de este objeto PROGRAM.
LastScanTime	DINT	GSV SSV	El tiempo necesario para ejecutar este programa desde la última vez que se ejecutó. El tiempo se representa en microsegundos.
MajorFaultRecord	DINT[11]	GSV SSV	Registra los fallos mayores para este programa Recomendamos que usted cree una estructura definida por el usuario para simplificar el acceso al atributo MajorFaultRecord:
Nombre:	**Tipo de datos:**	**Estilo:**	**Descripción:**
TimeLow	DINT	Decimal	los 32 bits inferiores de un valor de sello de hora de fallo
TimeHigh	DINT	Decimal	los 32 bits superiores de un valor de sello de hora de fallo
Type	INT	Decimal	tipo de fallo (programa, E/S, etc.)
Code	INT	Decimal	código único para el fallo (depende del tipo de fallo)
Info	DINT[8]	Hexadecimal	información acerca de un fallo específico (depende del tipo y código de fallo)
MaxScanTime	DINT	GSV SSV	El tiempo de ejecución máximo registrado para este programa. El tiempo se representa en microsegundos.
MinorFaultRecord	DINT[11]	GSV SSV	Registra los fallos menores para este programa Recomendamos que usted cree una estructura definida por el usuario para simplificar el acceso al atributo MinorFaultRecord:
Nombre:	**Tipo de datos:**	**Estilo:**	**Descripción:**
TimeLow	DINT	Decimal	los 32 bits inferiores de un valor de sello de hora de fallo
TimeHigh	DINT	Decimal	los 32 bits superiores de un valor de sello de hora de fallo
Type	INT	Decimal	tipo de fallo (programa, E/S, etc.)
Code	INT	Decimal	código único para el fallo (depende del tipo de fallo)
Info	DINT[8]	Hexadecimal	información acerca de un fallo específico (depende del tipo y código de fallo)
SFCRestart	INT	GSV SSV	no se usa – reservado para uso futuro

Cómo obtener acceso al objeto ROUTINE

El objeto ROUTINE proporciona información de estado acerca de la rutina. Especifique el nombre de la rutina para determinar el objeto ROUTINE deseado.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Instance	DINT	GSV	Proporciona el número de instancia de este objeto ROUTINE. Los valores válidos son 0 – 65,535.

Cómo obtener acceso al objeto SERIALPORT

El objeto SERIALPORT proporciona una interface al puerto de comunicación en serie.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
BaudRate	DINT	GSV	Especifica la velocidad en baudios. Los valores válidos son 110, 300, 600, 1,200, 2,400, 4,800, 9,600 y 19,200 (predeterminados).
DataBits	SINT	GSV	Especifica el número de bits de datos por carácter. **Valor: Significado:** 7 7 bits de datos (ASCII solamente) 8 8 bits de datos (opción predeterminada)
Paridad	SINT	GSV	Especifica la paridad. **Valor: Significado:** 0 sin paridad (sin paridad predeterminada) 1 imparidad (ASCII solamente) 2 paridad
RTSOffDelay	INT	GSV	La cantidad de tiempo necesario para retardar la desconexión de la línea RTS después de la transmisión del último carácter. El valor válido es 0 – 32,767. Retardo de conteos de períodos de 20 mseg. El valor predeterminado es 0 mseg.
RTSSendDelay	INT	GSV	La cantidad de tiempo necesario para retardar la transmisión del primer carácter de un mensaje después de activar la línea RTS. El valor válido es 0 – 32,767. Retardo de conteos de períodos de 20 mseg. El valor predeterminado es 0 mseg.
StopBits	SINT	GSV	Especifica el número de bits de paro. **Valor: Significado:** 1 1 bit de paro (opción predeterminada) 2 2 bits de paro (ASCII solamente)
PendingBaudRate	DINT	SSV	Valor pendiente para el atributo BaudRate.
PendingDataBits	SINT	SSV	Valor pendiente para el atributo DataBits.
PendingParity	SINT	SSV	Valor pendiente para el atributo Parity.
PendingRTSOffDelay	INT	SSV	Valor pendiente para el atributo RTSOffDelay.
PendingRTSSendDelay	INT	SSV	Valor pendiente para el atributo RTSSendDelay.
PendingStopBits	SINT	SSV	Valor pendiente para el atributo StopBits.

Para aplicar los valores para cualquiera de los atributos SERIALPORT pendientes:

1. Use una instrucción SSV para establecer el valor para el atributo pendiente.

   Usted puede establecer cualquier cantidad de atributos pendientes usando una instrucción SSV para cada atributo pendiente.

2. Use una instrucción MSG para aplicar el valor. La instrucción MSG aplica cada atributo pendiente establecido. Configure las instrucciones MSG como:

Ficha Configuración MSG:	Campo:	Valor:
Configuración	Tipo de mensaje	CIP genérico
	Código de servicio	0d hex
	Tipo de objeto	6f hex
	ID del objeto	1
	Atributo del objeto	deje en blanco
	Origen	deje en blanco
	Número de elementos	0
	Destino	deje en blanco
Comunicación	Ruta de acceso	ruta de comunicación a sí misma (1,*s* donde *s* = número de ranura del controlador)

Cómo obtener acceso al objeto TASK

El objeto TASK proporciona información de estado acerca de la tarea. Especifique el nombre de la tarea para determinar el objeto TASK deseado.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Instance	DINT	GSV	Proporciona el número de instancia de este objeto TASK. Los valores válidos son 0 – 31.
LastScanTime	DINT	GSV SSV	El tiempo necesario para ejecutar esta tarea desde la última vez que se ejecutó. El tiempo se representa en microsegundos.
MaxInterval	DINT[2]	GSV SSV	El intervalo de tiempo máximo entre las ejecuciones sucesivas de la tarea. DINT[0] contiene los bits inferiores del valor; DINT[1] contiene los 32 bits superiores del valor. Un valor de 0 indica 1 ó menos ejecuciones de la tarea.
MaxScanTime	DINT	GSV SSV	El tiempo de ejecución máximo registrado para este programa. El tiempo se representa en microsegundos.
MinxInterval	DINT[2]	GSV SSV	El intervalo de tiempo mínimo entre las ejecuciones sucesivas de la tarea. DINT[0] contiene los bits inferiores del valor; DINT[1] contiene los 32 bits superiores del valor. Un valor de 0 indica 1 ó menos ejecuciones de la tarea.
prioridad	INT	GSV	La prioridad relativa de esta tarea comparada con las otras tareas. Los valores válidos son 0 – 15.

Atributo:	Tipo de datos:	Instrucción:	Descripción:
Rate	DINT	GSV	El intervalo de tiempo entre las ejecuciones de la tarea. El tiempo se representa en microsegundos.
StartTime	DINT[2]	GSV SSV	El valor WALLCLOCKTIME cuando se inició la última ejecución de la tarea. DINT[0] contiene los bits inferiores del valor; DINT[1] contiene los 32 bits superiores del valor.
Temporizador de control (watchdog)	DINT	GSV SSV	El límite de tiempo para la ejecución de todos los programas asociados con esta tarea. El tiempo se representa en microsegundos. Si se introduce 0, se asignan estos valores: **Tiempo: Tipo de tarea:** 0.5 seg periódico 5.0 seg continuo

Cómo obtener acceso al objeto WALLCLOCKTIME

El objeto WALLCLOCKTIME proporciona un sello de hora que el controlador puede usar para el establecimiento de prioridades.

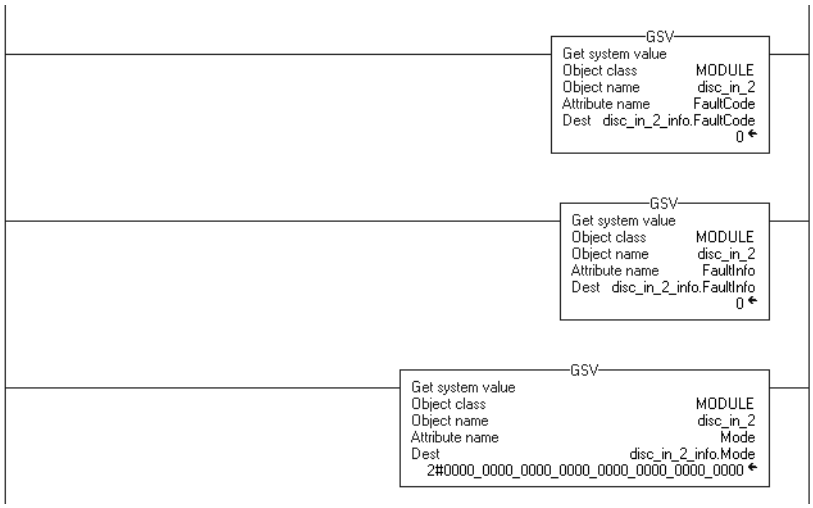
Atributo:	Tipo de datos:	Instrucción:	Descripción:
CSTOffset	DINT[2]	GSV SSV	El offset positivo del CurrentValue del objeto CST (hora coordinada del sistema); vea la página 3-38 DINT[0] contiene los bits inferiores del valor; DINT[1] contiene los 32 bits superiores del valor. El valor en µsegs. El valor predeterminado es 0.
CurrentValue	DINT[2]	GSV SSV	El valor actual de la hora del reloj. DINT[0] contiene los bits inferiores del valor; DINT[1] contiene los 32 bits superiores del valor. El valor es el número de microsegundos que han transcurrido desde 0000 hrs 1 de enero de 1972. Los objetos CST y WALLCLOCKTIME tienen una relación matemática con el controlador. Por ejemplo, si se suman CST CurrentValue y WALLCLOCKTIME CTSOffset, el resultado es WALLCLOCKTIME CurrentValue.
DateTime	DINT[7]	GSV SSV	La fecha y hora en un formato legible. DINT[0] año DINT[1] la representación de número entero del mes (1 – 12) DINT[2] la representación de número entero del mes (1 – 31) DINT[3] hora (0 – 23) DINT[4] minuto (0 – 59) DINT[5] segundos (0 – 59) DINT[6] microsegundos (0 – 999,999)

Ejemplo de programación GSV/SSV

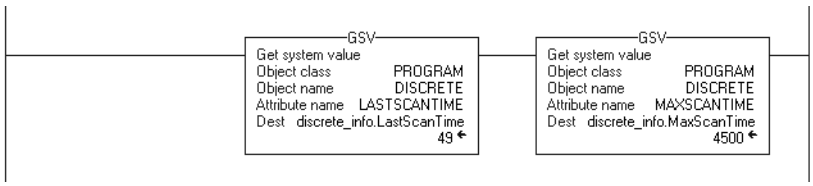
Cómo obtener información de fallo

Los ejemplos siguientes usan las instrucciones GSV para obtener información de fallo.

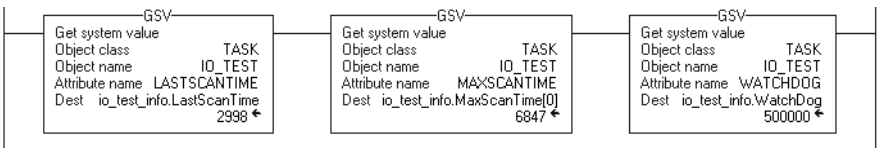
Cómo obtener información de fallo de E/S
Este ejemplo obtiene información de fallo desde el módulo de E/S *disc_in_2* y coloca los datos en una estructura definida por el usuario *disc_in_2_info*.



Cómo obtener información de estado del programa
Este ejemplo obtiene información de estado acerca del programa *discrete* y coloca los datos en una estructura definida por el usuario *discrete_info*.

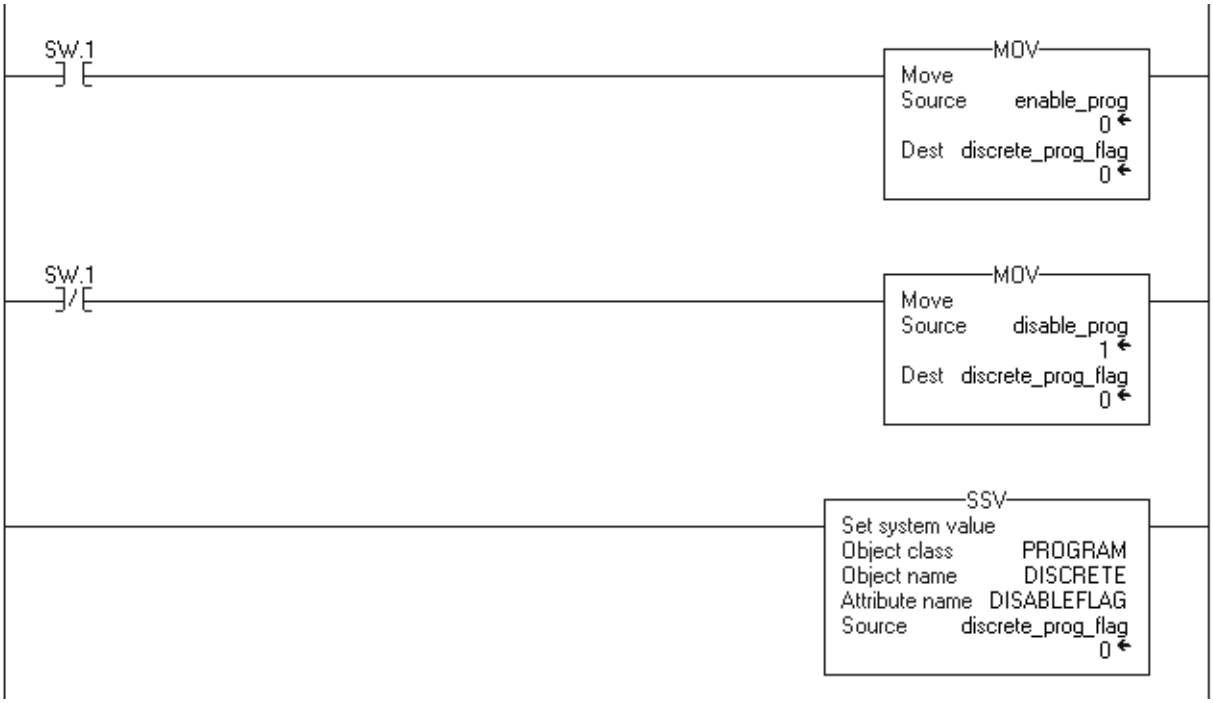


Cómo obtener información de estado de tarea
Este ejemplo obtiene información de estado acerca de la tarea *IO_test* y coloca los datos en una estructura definida por el usuario *io_test_info*.



Cómo establecer los indicadores de habilitación e inhabilitación

El ejemplo siguiente usa la instrucción SSV para habilitar o inhabilitar un programa. También se puede usar este método para habilitar o inhabilitar un módulo de E/S, lo cual es una solución de programa semejante al uso de bits de inhibición con un procesador PLC-5.



Según el estado de *SW.1*, coloque el valor apropiado en el atributo *disableflag* del programa *discrete*.

Notas:

Instrucciones de comparación
(CMP, EQU, GEQ, GRT, LEQ, LES, LIM, MEQ, NEQ)

Introducción

Las instrucciones de comparación le permiten comparar los valores usando una expresión o una instrucción de comparación específica.

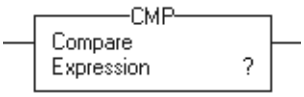
Si usted desea:	Use esta instrucción:	Vea la página:
comparar los valores según una expresión	CMP	4-2
determinar si dos valores son iguales	EQU	4-6
determinar si un valor es mayor o igual a otro valor	GEQ	4-8
determinar si un valor es mayor que otro valor	GRT	4-10
determinar si un valor es menor o igual a otro valor	LEQ	4-12
determinar si un valor es menor que otro valor	LES	4-14
determinar si un valor se encuentra entre dos otros valores	LIM	4-16
pasar dos valores a través de una máscara y determinar si son iguales	MEQ	4-19
determinar si un valor no es igual a otro valor	NEQ	4-22

Usted puede comparar los valores de diferentes tipos de datos, tales como el punto flotante (coma flotante) y números enteros.

Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Compare (CMP)

La instrucción CMP es una instrucción de entrada.



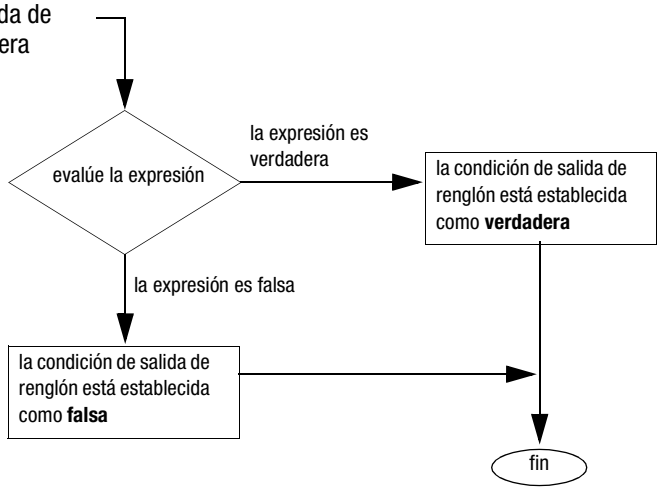
Operandos:

Operando:	Tipo:	Formato:	Descripción:
Expresión	SINT INT **DINT** **REAL**	inmediato tag	una expresión que consiste en tags y/o valores inmediatos separados por operadores.
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción CMP realiza una comparación de las operaciones aritméticas que se especifican en la expresión. Defina la expresión usando operadores, tags y valores inmediatos. Use los paréntesis ( ) para definir secciones de las expresiones más complejas.

La ejecución de una instrucción CMP es un poco más lenta y usa más memoria que la ejecución de las otras instrucciones de comparación. La ventaja de la instrucción CMP es que le permite introducir expresiones complejas en una sola instrucción.

Ejecución:

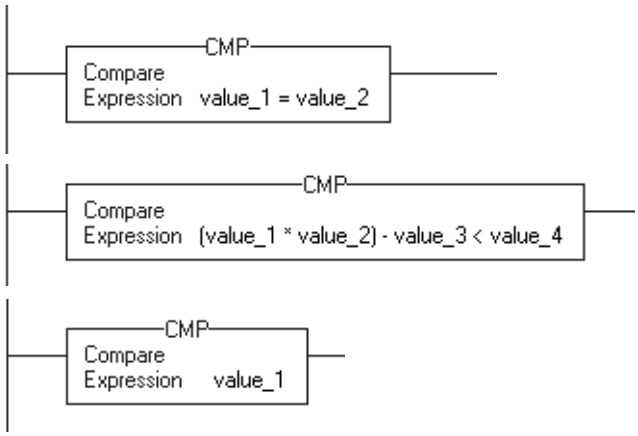
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición entrada de renglón es verdadera	

la condición entrada de renglón es verdadera
evalúe la expresión
la expresión es verdadera
la condición de salida de renglón está establecida como **verdadera**
la expresión es falsa
la condición de salida de renglón está establecida como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de CMP:



Si la instrucción CMP determina que la expresión es verdadera, la condición de salida de renglón se establece como verdadera.

Si introduce una expresión sin un operador de comparación, tal como *value_1 + value_2* ó *value_1*, la instrucción evalúa la expresión como:

Si la expresión es:	La condición de salida de renglón está establecida como:
sin cero	verdadero
cero	falso

Otros formatos:

Formato:	Sintaxis:
texto neutro	`CMP(`*`expression`*`);`
texto ASCII	`CMP` *`expression`*

Instrucciones relacionadas: CPT, FAL, FSC

Usted programa las expresiones en las instrucciones CMP de la misma manera que las expresiones en las instrucciones FSC. Use las secciones siguientes para obtener información acerca de operadores válidos, formato y orden de operación, los cuales son comunes en ambas instrucciones.

Operadores válidos

Operador:	Descripción:	Optimo:
+	sumar	DINT, REAL
–	restar/cambiar signo	DINT, REAL
*	multiplicar	DINT, REAL
/	dividir	DINT, REAL
=	igual	DINT, REAL
<	menor que	DINT, REAL
<=	menor que o igual	DINT, REAL
>	mayor que	DINT, REAL
>=	mayor que o igual	DINT, REAL
<>	diferente	DINT, REAL
**	exponente (x a y)	DINT, REAL
ABS	valor absoluto	DINT, REAL
ACS	arco coseno	real
AND	función Y	DINT
ASN	arco seno	REAL
ATN	arco tangente	REAL

Operador:	Descripción:	Optimo:
COS	coseno	REAL
DEG	radianes a grados	DINT, REAL
FRD	BCD a número entero	DINT
LN	logaritmo natural	REAL
LOG	logaritmo de base 10	REAL
MOD	módulo-división	DINT, REAL
NOT	complemento bit a bit	DINT
OR	función O	DINT
RAD	radianes a grados	DINT, REAL
SIN	seno	REAL
SQR	raíz cuadrada	DINT, REAL
TAN	tangente	REAL
TOD	número entero a BCD	DINT
TRN	truncar	DINT, REAL
XOR	OR exclusivo, bit a bit	DINT

Cómo formatear expresiones

Para cada operador que usted usa en una expresión, tiene que proporcionar uno o dos operandos (tags o valores inmediatos). Use la tabla siguiente para formatear los operadores y operandos dentro de una expresión:

Para los operadores que realizan la operación en:	Use este formato:	Ejemplos:
un operando	operador (operando)	ABS(*tag_a*)
dos operandos	operand_a operador operand_b	• *tag_b* + 5 • *tag_c* AND *tag_d* • (*tag_e* ** 2) MOD (*tag_f* / *tag_g*)

Cómo determinar el orden de operación

Las operaciones que usted escribe en la expresión son realizadas por la instrucción en un orden determinado y no necesariamente según el orden en que las escribió. Puede anular el orden de operación agrupando los términos entre paréntesis, lo cual causa que la instrucción realice una operación entre paréntesis antes que las otras operaciones.

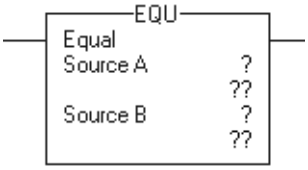
Las operaciones de orden igual se realizan desde la izquierda hacia la derecha.

Orden:	Operación:
1.	( )
2.	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3.	**
4.	– (cambiar signo), NOT
5.	*, /, MOD
6.	<, <=, >, >=, =
7.	– (restar), +
8.	AND
9.	XOR
10.	OR

Equal to (EQU)

La instrucción EQU es una instrucción de entrada.

Operandos:

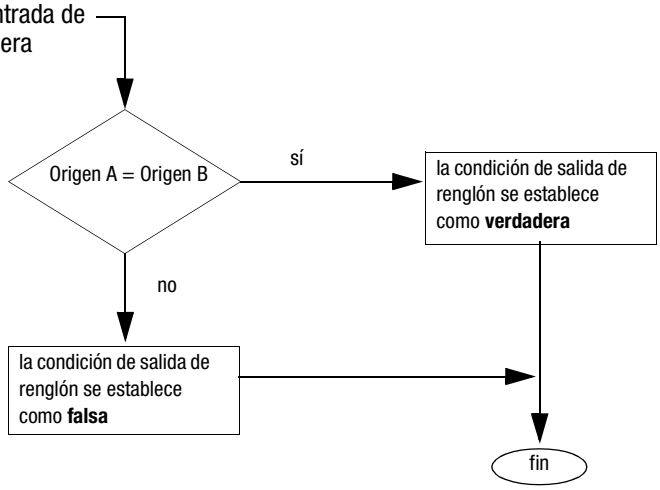


Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **real**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **real**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción EQU determina si el origen A es igual al origen B.

Los valores REAL rara vez son absolutamente iguales. Si usted necesita determinar la igualdad de dos valores REAL, use la instrucción LIM.

Ejecución:

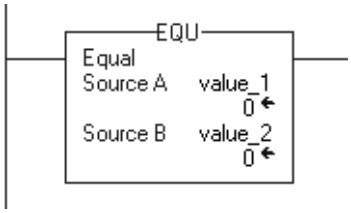
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Origen A = Origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de EQU:



Si *value_1* es igual que *value_2*, la condición de salida de renglón se establece como verdadera.

Otros formatos:

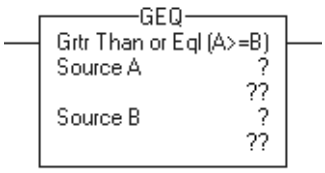
Formato:	Sintaxis:
texto neutro	`EQU(source_A,source_B);`
texto ASCII	`EQU source_A source_B`

Instrucciones relacionadas: CMP, GEQ, LEQ, MEQ, NEQ

Greater Than or Equal to (GEQ)

La instrucción GEQ es una instrucción de entrada.

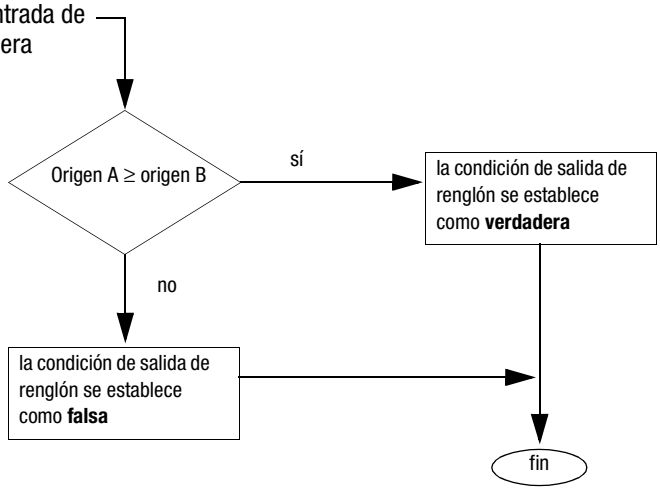
Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción GEQ determina si el origen A es mayor o igual que el origen B.

Ejecución:

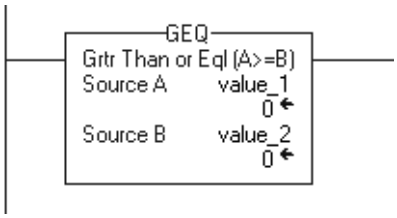
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón está establecida como falsa.
la condición de entrada de renglón es verdadera	

la condición de entrada de renglón es verdadera
Origen A ≥ origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de GEQ:



Si *value_1* es mayor o igual que *value_2*, la condición de salida de renglón se establece como verdadera.

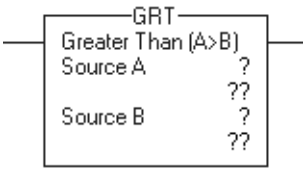
Otros formatos:

Formato:	Sintaxis:
texto neutro	`GEQ(source_A,source_B);`
texto ASCII	`GEQ source_A source_B`

Instrucciones relacionadas: CMP, EQU, LEQ, MEQ, NEQ

Greater Than (GRT)

La instrucción GRT es una instrucción de entrada.

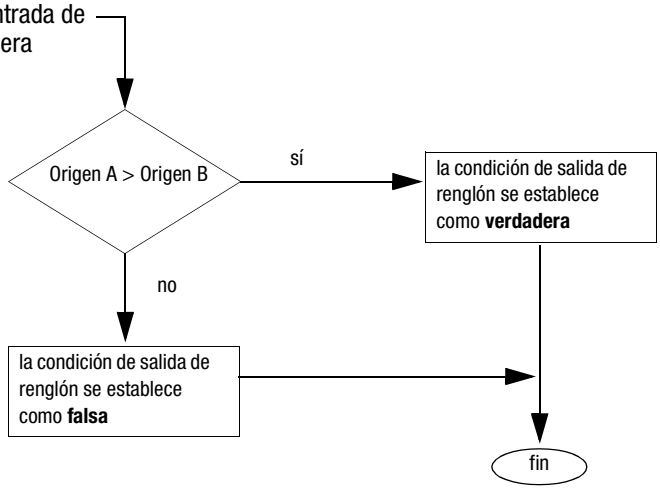


Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción GRT determina si el origen A es mayor que el origen B.

Ejecución:

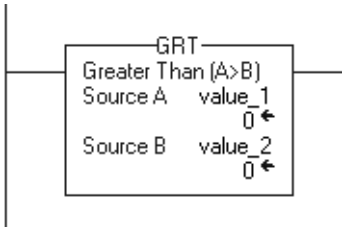
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Origen A > Origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de GRT:



Si *value_1* es mayor que *value_2*, la instrucción es verdadera.

Otros formatos:

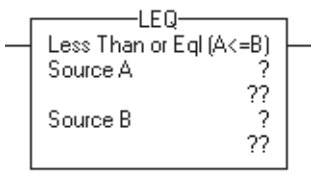
Formato:	Sintaxis:
texto neutro	`GRT(source_A,source_B);`
texto ASCII	`GRT source_A source_B`

Instrucciones relacionadas: CMP, LES

Less Than or Equal to (LEQ)

La instrucción LEQ es una instrucción de entrada.

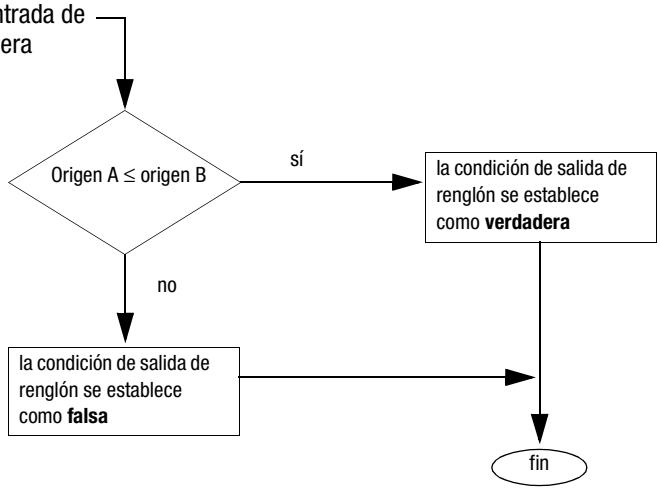
Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción LEQ determina si el origen A es menor que o igual al origen B.

Ejecución:

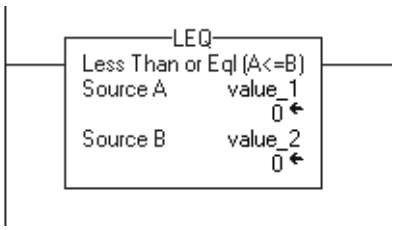
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

la condición de entrada de renglón es verdadera
Origen A ≤ origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de LEQ:



Si *value_1* es menor o igual que *value_2*, la condición de salida de renglón se establece como verdadera.

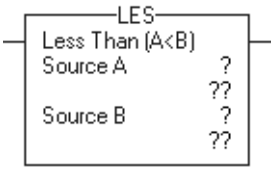
Otros formatos:

Formato:	Sintaxis:
texto neutro	`LEQ(source_A,source_B);`
texto ASCII	`LEQ source_A source_B`

Instrucciones relacionadas: CMP, EQU, GEQ, MEQ, NEQ

Less Than (LES)

La instrucción LES es una instrucción de entrada.

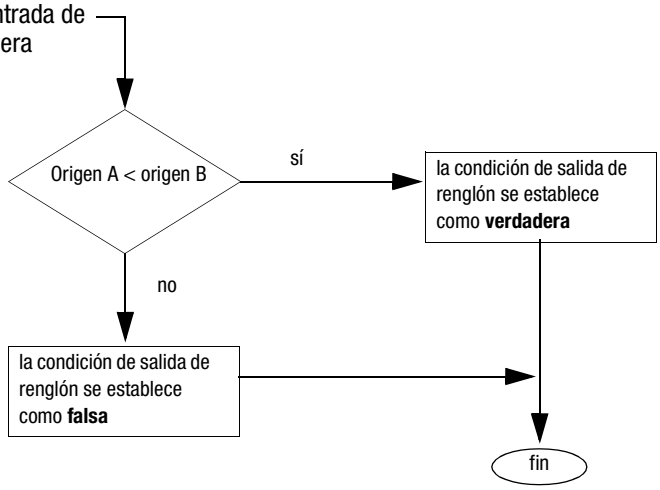


Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción LES determina si el origen A es menor que el origen B.

Ejecución:

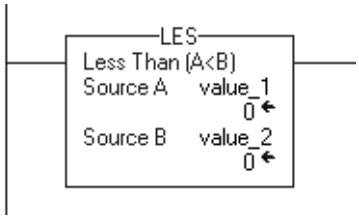
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón está establecida como falsa.
la condición de entrada de renglón es verdadera	

Origen A < origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de LES:



Si *value_1* es menor que *value_2*, la condición de salida de renglón se establece como verdadera.

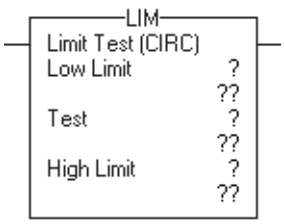
Otros formatos:

Formato:	Sintaxis:
texto neutro	`LES(source_A,source_B);`
texto ASCII	`LES source_A source_B`

Instrucciones relacionadas: CMP, GRT

Limit (LIM)

La instrucción LIM es una instrucción de entrada.



Operandos:

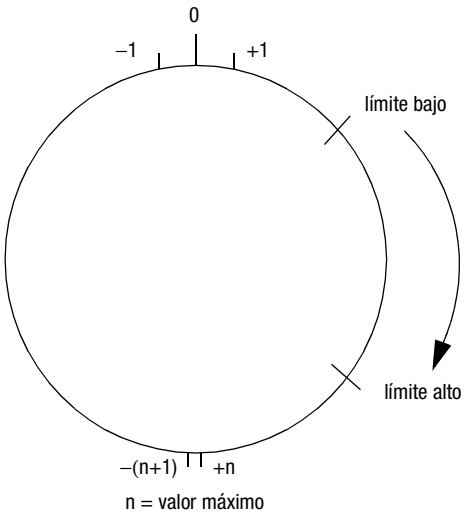
Operando:	Tipo:	Formato:	Descripción:
Límite bajo	SINT INT **DINT** **REAL**	inmediato tag	valor del límite inferior
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Prueba	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Límite alto	SINT INT **DINT** **REAL**	inmediato tag	valor del límite superior
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción LIM determina si el valor de prueba se encuentra dentro del rango de los límites inferior y superior.

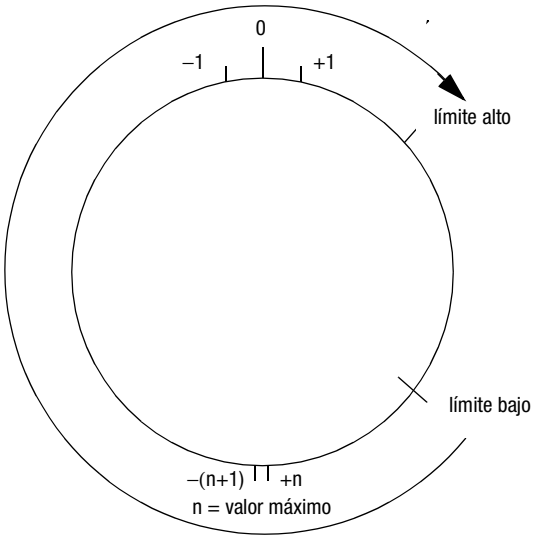
Los números enteros con signo continúan del número positivo máximo al número negativo máximo cuando se establece el bit más significativo. Por ejemplo, en los números enteros de 16 bits (tipo INT), el número entero positivo máximo es 32,767, el cual se representa en hexadecimal como 16#7FFF (todos los bits de 0 a 14 están establecidos). Si usted incrementa dicho número en un valor de 1, el resultado es 16#8.000 (el bit 16 está establecido). Para los números enteros con signo, 16#8.000 hexadecimal es igual a –32,768 decimal. El incremento desde este punto hasta que se establecen los 16 bits resulta en 16#FFF, el cual es igual a –1 decimal.

Esto se puede mostrar como una línea numérica circular (vea los diagramas a continuación). La instrucción LIM comienza a partir del límite inferior e incrementa hacia la derecha hasta que llega al límite superior. Cualquier valor de prueba en el rango horario desde el límite inferior al límite superior establece la condición de salida de renglón como verdadera. Cualquier valor de prueba en el rango horario desde el límite superior al límite inferior establece la condición de salida de renglón como verdadera.

Límite inferior ≤ límite superior
La instrucción es verdadera si el valor de prueba es igual a o se encuentra entre los límites inferior y superior.

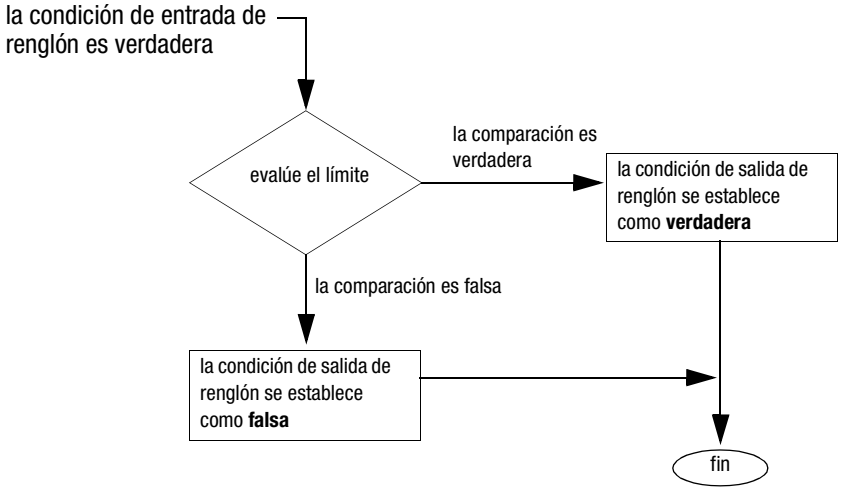


Límite inferior ≥ límite superior
La instrucción es verdadera si el valor de prueba es igual a o se encuentra fuera de los límites inferior y superior.



Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.


la condición de entrada de renglón es verdadera
evalúe el límite
la comparación es verdadera
la condición de salida de renglón se establece como **verdadera**
la comparación es falsa
la condición de salida de renglón se establece como **falsa**
fin

Si es límite inferior:	Y el valor de prueba es:	La condición de salida de renglón está establecida como:
≤ límite alto	igual a o entre los límites	verdadero
	no igual a o fuera de los límites	falso
≥ límite alto	igual a o fuera de los límites	verdadero
	no igual a o dentro de los límites	falso

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo LIM:

ejemplo 1

Límite inferior ≤ límite superior
Cuando 0 ≤ *value* ≥ 100, se enciende light_*1*.

ejemplo 2

Límite inferior ≥ límite superior
Cuando *value* ≥ 0 ó *value* ≤ –100, se enciende light_*1* turns.

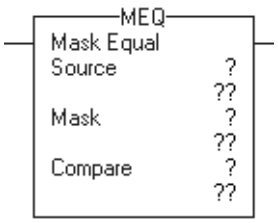
Otros formatos:

Formato:	Sintaxis:
texto neutro	`LIM(low_limit,test,high_limit);`
texto ASCII	`LIM low_limit test high_limit`

Instrucciones relacionadas: CMP

Mask Equal to (MEQ)

La instrucción MEQ es una instrucción de entrada.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT**	inmediato tag	valor que se prueba contra la comparación
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Máscara	SINT INT **DINT**	inmediato tag	qué bits se bloquean o se pasan
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Comparación	SINT INT **DINT**	inmediato tag	valor que se prueba contra el origen
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción MEQ pasa los valores de origen y comparación a través de una máscara y compara los resultados.

Un número “1” en la máscara significa que se pasa el bit de datos. Un número “0” en la máscara significa que se bloquea el bit de datos. Típicamente los valores de origen, máscara y comparación son todos del mismo tipo de datos.

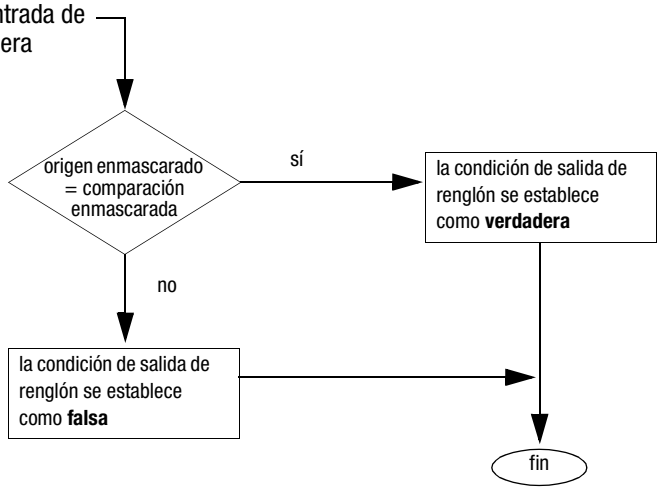
Si combina los tipos de datos enteros, la instrucción llena los bits superiores de los tipos de datos enteros menores con ceros para que tengan el mismo tamaño que el tipo de datos más grande.

Cómo introducir un valor de máscara inmediato

Cuando introduce una máscara, el software de programación tiene valores decimales predeterminados. Si desea introducir una máscara usando otro formato, ponga el prefijo correcto ante el valor.

Prefijo:	Descripción:
16#	hexadecimal por ejemplo; 16#0F0F
8#	octal por ejemplo; 8#16
2#	binario por ejemplo; 2#00110011

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

origen enmascarado = comparación enmascarada
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo MEQ:

ejemplo 1

El *value_1* enmascarado es igual al *value_2* enmascarado, por lo tanto se enciende *light_1*. Un número 0 en la máscara impide que la instrucción compare el bit (representado por x en el ejemplo).

ejemplo 2

value_1	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1
mask_1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
value_1 enmascarado	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1

value_2	0	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0
mask_1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
value_2 enmascarado	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

El *value_1* enmascarado no es igual al *value_2* enmascarado, por lo tanto se apaga *light_1*. Un número 0 en la máscara impide que la instrucción compare el bit (representado por x en el ejemplo).

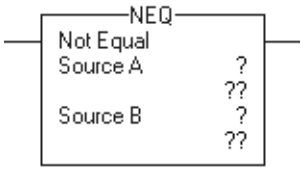
Otros formatos:

Formato:	Sintaxis:
texto neutro	`MEQ(source,mask,compare);`
texto ASCII	`MEQ source mask compare`

Instrucciones relacionadas: CMP, EQU, GEQ, LEQ, MEQ, NEQ

Not Equal to (NEQ)

La instrucción NEQ es una instrucción de entrada.

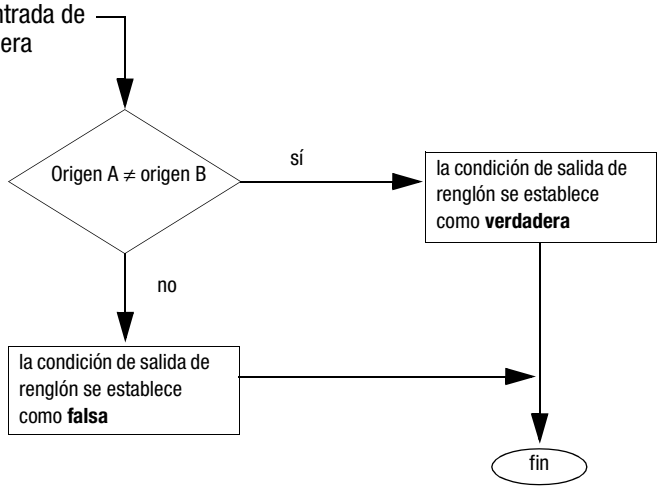


Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	inmediato tag	valor que se prueba contra el origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción NEQ determina si dos valores no son iguales.

Ejecución:

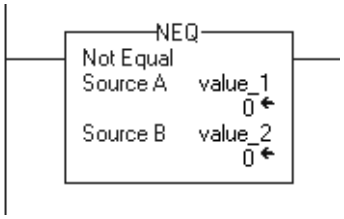
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón está establecida como falsa.
la condición de entrada de renglón es verdadera	

Origen A ≠ origen B
sí
la condición de salida de renglón se establece como **verdadera**
no
la condición de salida de renglón se establece como **falsa**
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de NEQ:



Si *value_1* no es igual a *value_2*, la condición de salida de renglón se establece como verdadera.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`NEQ(source_A,source_B);`
texto ASCII	`NEQ source_A source_B`

Instrucciones relacionadas: CMP, EQU, LEQ, GEQ, MEQ

Notas:

Instrucciones de cálculo/matemáticas
(CPT, ADD, SUB, MUL, DIV, MOD, SQR, NEG, ABS)

Introducción

Las instrucciones de cálculo/matemáticas evalúan las operaciones aritméticas usando una expresión o una instrucción aritmética específica.

Si usted desea:	Use esta instrucción:	Vea la página:
evaluar una expresión	CPT	5-2
sumar dos valores	ADD	5-5
restar dos valores	SUB	5-7
multiplicar dos valores	MUL	5-9
dividir dos valores	DIV	5-11
determinar el resto después de dividir un valor entre otro	MOD	5-13
calcular la raíz cuadrada de un valor	SQR	5-15
hallar el signo opuesto de un valor	NEG	5-17
hallar el valor absoluto de un valor	ABS	5-18

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

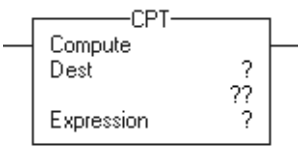
Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Una instrucción de cálculo/matemática se ejecuta cada vez que se escanea la instrucción siempre que la condición de entrada de renglón sea verdadera. Si desea que la expresión se evalúe solamente una vez, use cualquier instrucción de un impulso para activar la instrucción.

Compute (CPT)

La instrucción CPT es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado
Expresión	SINT INT **DINT** **REAL**	valor inmediato tag	una expresión que consiste en tags y/o valores inmediatos separados por operadores.
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Descripción: La instrucción CPT realiza las operaciones aritméticas que se definen en la expresión. Cuando está habilitada, la instrucción CPT evalúa la expresión y coloca el resultado en el destino.

La ejecución de una instrucción CPT es un poco más lenta y usa más memoria que la ejecución de las otras instrucciones de cálculo/matemáticas. La ventaja de la instrucción CPT es que le permite introducir expresiones complejas en una sola instrucción.

▶ La longitud de una expresión es *ilimitada*.

Ejecución:

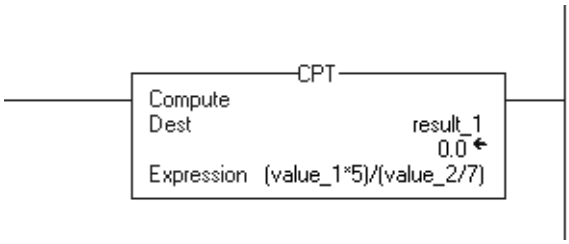
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción CPT evalúa la expresión y coloca el resultado en el destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

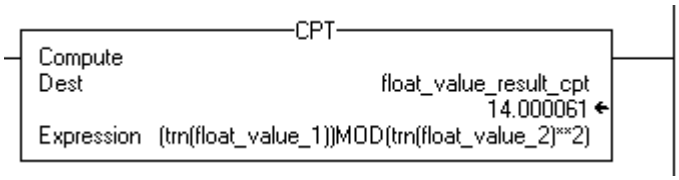
Ejemplos de CPT:

ejemplo 1



Cuando está habilitada, la instrucción CPT evalúa *value_1* multiplicado por 5 y divide el resultado entre el resultado de *value_2* dividido entre 7 y coloca el resultado final en result_*1*.

ejemplo 2



Cuando está habilitada, la instrucción CPT trunca *float_value_1* y *float_value_2*, aumenta el *float_value_2* truncado a la potencia de dos y divide el *float_value_1* truncado entre el resultado, y almacena el resto después de la división en *float_value_result_cpt*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`CPT(destination,expression);`
texto ASCII	`CPT destination expression`

Instrucciones relacionadas: CMP, FAL, FSC

Usted programa las expresiones en las instrucciones CPT de la misma manera que las expresiones en las instrucciones FAL. Use las secciones siguientes para obtener información acerca de operadores válidos, formato y orden de operación, los cuales son comunes en ambas instrucciones.

Operadores válidos

Operador:	Descripción:	Optimo:
+	sumar	DINT, REAL
–	restar/cambiar signo	DINT, REAL
*	multiplicar	DINT, REAL
/	dividir	DINT, REAL
**	exponente (x a y)	DINT, REAL
ABS	valor absoluto	DINT, REAL
ACS	arco coseno	REAL
AND	función Y	DINT
ASN	arco seno	REAL
ATN	arco tangente	REAL
COS	coseno	REAL
DEG	radianes a grados	DINT, REAL
FRD	BCD a número entero	DINT

Operador:	Descripción:	Optimo:
LN	logaritmo natural	REAL
LOG	logaritmo de base 10	REAL
MOD	módulo-división	DINT, REAL
NOT	complemento bit a bit	DINT
OR	función O	DINT
RAD	radianes a grados	DINT, REAL
SIN	seno	REAL
SQR	raíz cuadrada	DINT, REAL
TAN	tangente	REAL
TOD	número entero a BCD	DINT
TRN	truncar	DINT, REAL
XOR	OR exclusivo, bit a bit	DINT

Cómo formatear expresiones

Para cada operador que usted usa en una expresión, tiene que proporcionar uno o dos operandos (tags o valores inmediatos). Use la tabla siguiente para formatear los operadores y operandos dentro de una expresión:

Para los operadores que realizan la operación en:	Use este formato:	Ejemplos:
un operando	operador (operando)	ABS(*tag_a*)
dos operandos	operand_a operador operand_b	• *tag_b* + 5 • *tag_c* AND *tag_d* • (*tag_e* ** 2) MOD (*tag_f* / *tag_g*)

Cómo determinar el orden de operación

Las operaciones que usted escribe en la expresión son realizadas por la instrucción en un orden determinado y no necesariamente según el orden en que las escribió. Puede anular el orden de operación agrupando los términos entre paréntesis, lo cual causa que la instrucción realice una operación entre paréntesis antes que las otras operaciones.

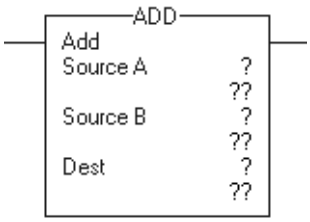
Las operaciones de orden igual se realizan desde la izquierda hacia la derecha.

Orden:	Operación:
1.	( )
2.	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3.	**
4.	– (cambiar signo), NOT
5.	*, /, MOD
6.	– (restar), +
7.	AND
8.	XOR
9.	OR

Add (ADD)

La instrucción ADD es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	valor inmediato tag	valor que se suma al origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	valor inmediato tag	valor que se suma al origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción ADD suma el origen A al origen B y coloca el resultado en el destino.

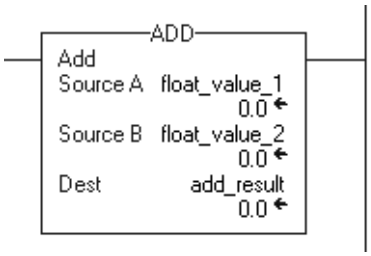
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = origen A + origen B La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de ADD:



Una vez habilitada, la instrucción ADD suma *float_value_1* a *float_value_2* y coloca el resultado en *add_result*.

Otros formatos:

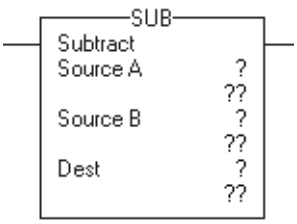
Formato:	Sintaxis:
texto neutro	`ADD(source_A,source_B,destination);`
texto ASCII	`ADD source_A source_B destination`

Instrucciones relacionadas: CPT, DIV, MUL, SUB

Subtract (SUB)

La instrucción SUB es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	valor inmediato tag	valor del cual se resta el origen B
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	valor inmediato tag	valor que se resta del origen A
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción SUB resta el origen B del origen A y coloca el resultado en el destino.

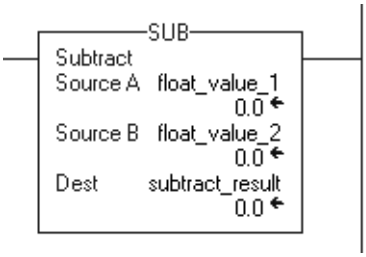
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = origen A – origen B La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de SUB:



Una vez habilitada, la instrucción SUB resta *float_value_2* de *float_value_1* y coloca el resultado en *subtract_result*.

Otros formatos:

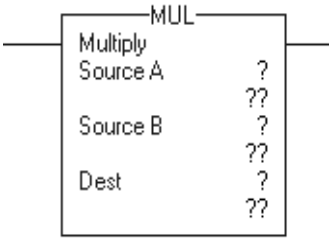
Formato:	Sintaxis:
texto neutro	`SUB(source_A,source_B,destination);`
texto ASCII	`SUB source_A source_B destination`

Instrucciones relacionadas: CPT, ADD DIV, MUL

Multiply (MUL)

La instrucción MUL es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	valor inmediato tag	valor del multiplicando
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	valor inmediato tag	valor del multiplicador
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción MUL multiplica el origen A por el origen B y coloca el resultado en el destino.

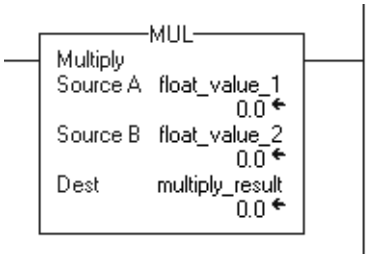
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = Origen A ∗ Origen B La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de MUL:



Una vez habilitada, la instrucción MUL multiplica *float_value_1* por *float_value_2* y coloca el resultado en *multiply_result*.

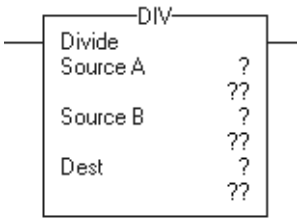
Otros formatos:

Formato:	Sintaxis:
texto neutro	`MUL(source_A,source_B,destination);`
texto ASCII	`MUL source_A source_B destination`

Instrucciones relacionadas: CPT, ADD, DIV, SUB

Divide (DIV)

La instrucción DIV es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	valor inmediato tag	valor del dividendo
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	valor inmediato tag	valor del divisor
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción:

La instrucción DIV divide el origen A entre el origen B y coloca el resultado en el destino. Si el destino *no* es REAL, la instrucción procesa la porción fraccionaria del resultado de la manera siguiente:

Si el origen A:	La porción fraccionaria del resultado:	Ejemplo:		
y el origen B *no* son REAL	se trunca	Origen A	DINT	5
		Origen B	DINT	3
		Destino	DINT	1
o el origen B es REAL	se redondea	Origen A	REAL	5.0
		Origen B	DINT	3
		Destino	DINT	2

Si el origen B (el divisor) es cero, el destino se establece como igual al origen B (el dividendo) y se registra un fallo menor como un overflow aritmético. Puede detectar la posible ocurrencia de una operación de división entre cero monitoreando el bit de fallo menor (S:MINOR).

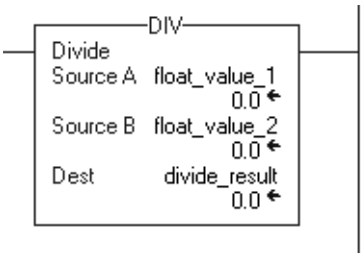
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = origen A/origen B La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de DIV:



Una vez habilitada, la instrucción DIV divide *float_value_1* entre *float_value_2* y coloca el resultado en *divide_result*.

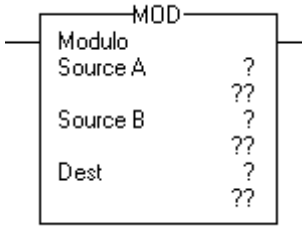
Otros formatos:

Formato:	Sintaxis:
texto neutro	`DIV(source_A,source_B,destination);`
texto ASCII	`DIV source_A source_B destination`

Instrucciones relacionadas: CPT, ADD, MUL, SUB

Modulo (MOD)

La instrucción MOD es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT** **REAL**	valor inmediato tag	valor del dividendo
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT** **REAL**	valor inmediato tag	valor del divisor
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción:

La instrucción MOD divide el origen A entre el origen B y coloca el resto en el destino.

Si el origen B (el divisor) es cero, se registra un fallo menor como un overflow aritmético y el destino se establece según lo indicado en la tabla siguiente:

Si el divisor es cero y el destino es un tag de este tipo de datos:	El destino está establecido en:
SINT, INT o DINT	cero
REAL	infinito

▶ Para detectar la posible ocurrencia de una operación de división entre cero, examine el bit de fallo menor (S:MINOR).

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establecea como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = Origen A – (TRN (Origen A / Origen B) * Origen B) La condición de salida de renglón se establece como verdadera.

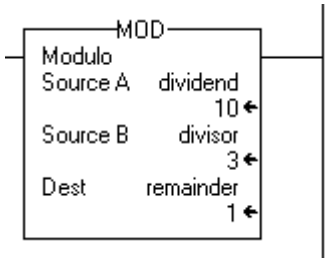
Indicadores de estado aritmético:

Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurre un fallo menor si:	Tipo de fallo:	Código de fallo:
el divisor es cero	4	4

Ejemplo de MOD:



Cuando está habilitada, la instrucción MOD divide el *dividendo* entre el *divisor* y coloca el resto en *resto*. En este ejemplo, tres se divide entre 10 tres veces, con un resto de uno.

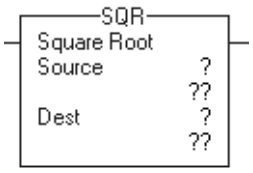
Otros formatos:

Formato:	Sintaxis:
texto neutro	`MOD(source_A,source_B,destination);`
texto ASCII	`MOD source_A source_B destination`

Instrucciones relacionadas: CPT, ADD, MUL, SUB, DIV

Square Root (SQR)

La instrucción SQR es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT** **REAL**	valor inmediato tag	hallar la raíz cuadrada de este valor
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción SQR calcula la raíz cuadrada del origen y coloca el resultado en el destino. Si el destino *no* es REAL, la instrucción procesa la porción fraccionaria del resultado de la manera siguiente:

Si el origen:	La porción fraccionaria del resultado:	Ejemplo:		
no es REAL	se trunca	Origen	DINT	3
		Destino	DINT	1
es REAL	se redondea	Origen	REAL	3.0
		Destino	DINT	2

Si el origen es negativo, la instrucción halla el valor absoluto del origen antes de calcular la raíz cuadrada.

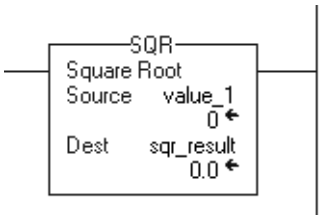
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	$Destino = \sqrt{Origen}$ La condición de salida de renglón se establece como verdadera

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de SQR:



Una vez habilitada, la instrucción SQR calcula la raíz cuadrada de *value_1* y coloca el resultado en *sqr_result*.

Otros formatos:

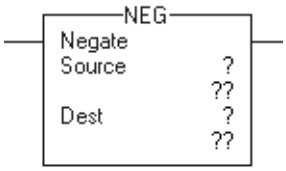
Formato:	Sintaxis:
texto neutro	`SQR(source,destination);`
texto ASCII	`SQR source destination`

Instrucciones relacionadas: CPT, NEG

Negate (NEG)

La instrucción NEG es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT** **REAL**	valor inmediato tag	valor del cual se cambia el signo
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción NEG cambia el signo del origen y coloca el resultado en el destino. Si cambia el signo de un valor negativo, el resultado es positivo. Si cambia el signo de un valor positivo, el resultado es negativo.

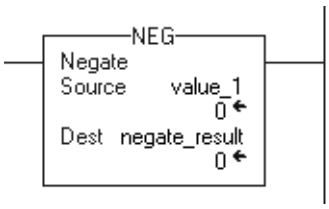
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = 0 – Origen La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de NEG:



Una vez habilitada, la instrucción NEG cambia el signo de *value_1* y coloca el resultado en *negate_result*.

Otros formatos:

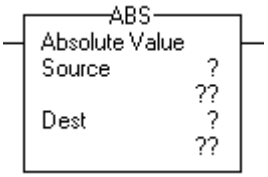
Formato:	Sintaxis:
texto neutro	`NEG(source,destination);`
texto ASCII	`NEG source destination`

Instrucciones relacionadas: CPT, SQR

Absolute Value (ABS)

La instrucción ABS es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT** **REAL**	valor inmediato tag	valor desde el cual se extrae el valor absoluto
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción ABS halla el valor absoluto del origen y coloca el resultado en el destino.

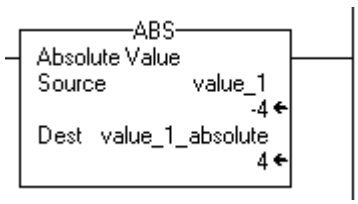
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Destino = | Origen | La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de ABS:



Cuando está habilitada, la instrucción ABS coloca el valor absoluto de *value_1* en *value_1_absolute*. En este ejemplo, el valor absoluto de cuatro negativo es cuatro positivo.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`ABS(source,destination);`
texto ASCII	`ABS source destination`

Instrucciones relacionadas: CPT, NEG, SQR, TRN

Notas:

Instrucciones de transferencia/lógica
(MOV, MVM, BTD, CLR, AND, OR, XOR, NOT)

Introducción

Las instrucciones de transferencia modifican y transfieren los bits.

Si usted desea:	Use esta instrucción:	Vea la página:
copiar un valor	MOV	6-2
copiar una porción especifica de un número entero	MVM	6-3
mover bits dentro de un número entero o entre números enteros	BTD	6-5
borrar un valor	CLR	6-8

Las instrucciones lógicas realizan operaciones lógicas en los bits.

Si usted desea realizar una:	Use esta instrucción:	Vea la página:
función A, bit a bit	AND	6-9
función O, bit a bit	OR	6-11
función O exclusivo, bit a bit	XOR	6-13
función NO, bit a bit	NOT	6-15

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

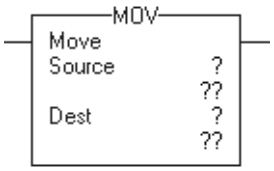
Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Una instrucción de transferencia/lógica se ejecuta cada vez que se escanea la instrucción siempre que la condición de entrada de renglón sea verdadera. Si desea que la expresión se evalúe solamente una vez, use cualquier instrucción de un impulso para activar la instrucción de transferencia/ lógica.

Move (MOV)

La instrucción MOV es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT** **REAL**	valor inmediato tag	valor que se mueve (se copia)
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción MOV copia el origen al destino. El origen no se cambia.

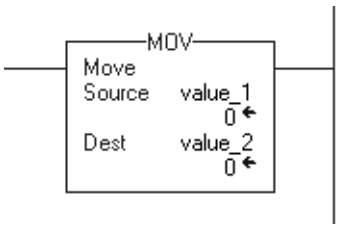
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción copia el origen en el destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de MOV:



Una vez habilitada, la instrucción MOV copia los datos en *value_1* a *value_2*.

Otros formatos:

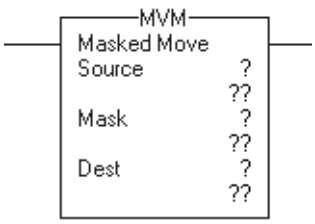
Formato:	Sintaxis:
texto neutro	`MOV(source,destination);`
texto ASCII	`MOV source destination`

Instrucciones relacionadas: BTD, CLR, MVM

Masked Move (MVM)

La instrucción MVM es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT**	valor inmediato tag	valor que se mueve
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Máscara	SINT INT **DINT**	valor inmediato tag	qué bits se bloquean o se pasan
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT**	tag	tag para almacenar el resultado

Descripción: La instrucción MVM copia el origen a un destino y permite que se enmascaren porciones de los datos. El origen no se cambia.

Una vez habilitada, la instrucción MVM usa una máscara para pasar o bloquear los bits de datos del origen. Un número “1” en la máscara significa que se pasa el bit de datos. Un número “0” en la máscara significa que se bloquea el bit de datos.

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

Cómo introducir un valor de máscara inmediato

Cuando introduce una máscara, el software de programación vuelve a establecer los valores decimales predeterminados. Si desea introducir una máscara usando otro formato, ponga el prefijo correcto antes del valor.

Prefijo:	Descripción:
16#	hexadecimal por ejemplo; 16#0F0F
8#	octal por ejemplo; 8#16
2#	binario por ejemplo; 2#00110011

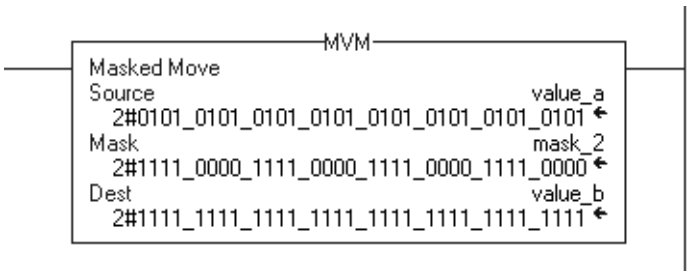
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucciones pasa el origen a través de la máscara y copia el resultado en el destino. Los bits sin máscara en el destino no se cambian. La condición de salida de renglón se establece como verdadera.

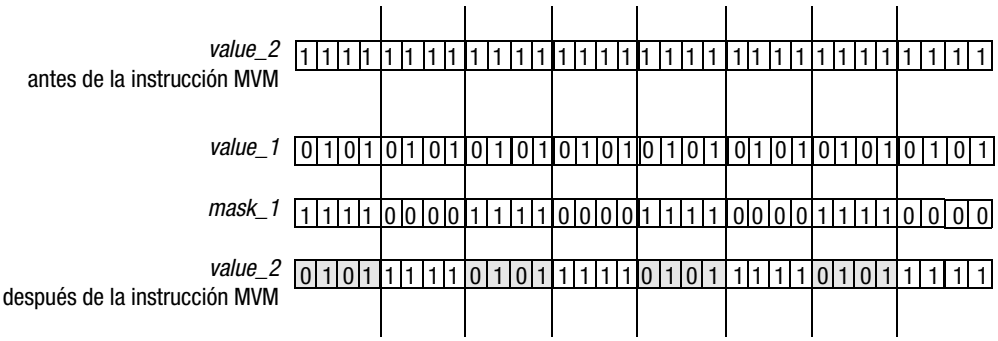
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de MVM:



Una vez habilitada, la instrucción MVM copia los datos de *value_a* a *value_b*, permitiendo así que los datos se enmascaren (un 0 enmascara los datos en *value_a*).



Las casillas sombreadas muestran los bits que se cambiaron en *value_2*.

Otros formatos:

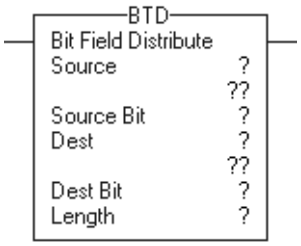
Formato:	Sintaxis:
texto neutro	`MVM(source,mask,destination);`
texto ASCII	`MVM source mask destination`

Instrucciones relacionadas: BTD, CLR, MOV

Bit Field Distribute (BTD)

La instrucción BTD es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT**	valor inmediato tag	tag que contiene los bits que se mueven
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Bit de origen	DINT	valor inmediato (0 – 31 DINT) (0 – 15 INT) (0 – 7 SINT)	número del bit (el número de bit menor de todos) desde el cual se inicia el movimiento debe encontrarse dentro del rango válido para el tipo de datos del origen
Destino	SINT INT **DINT**	tag	tag al cual se mueven los bits
Bit de destino	DINT	valor inmediato (0 – 31 DINT) (0 – 15 INT) (0 – 7 SINT)	el número del bit (el número de bit menor de todos) donde se inicia el copiado de los bits del origen debe encontrarse dentro del rango válido para el tipo de datos del destino
Longitud	DINT	inmediato (1 – 32)	número de bits que se mueven

Descripción:

La instrucción BTD copia los bits especificados desde el origen, desplaza los bits en la posición apropiada y escribe los bits al destino. No se cambia el resto del destino.

Una vez habilitada, la instrucción BTD copia un grupo de bits desde el origen hacia el destino. El grupo de bits se identifica por el bit de origen (el número de bit menor del grupo) y la longitud (el número de bits que se copian). El bit de destino identifica el número de bit menor con el cual se comienza en el destino. El origen no se cambia.

Si la longitud del campo de bits se extiende más allá del destino, la instrucción no guarda los bits adicionales. Los bits adicionales no pasan a la próxima palabra.

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

Ejecución:

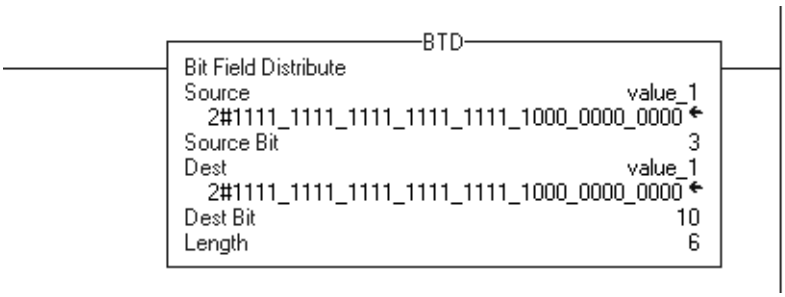
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción copia y desplaza los bits de origen al destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

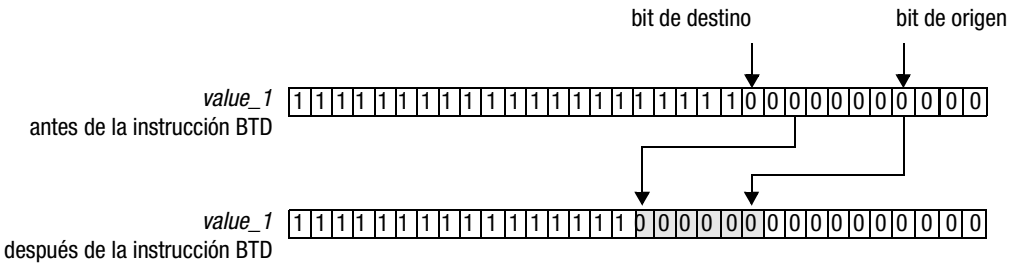
Condiciones de fallo: ninguna

Ejemplo de BTD:

ejemplo 1

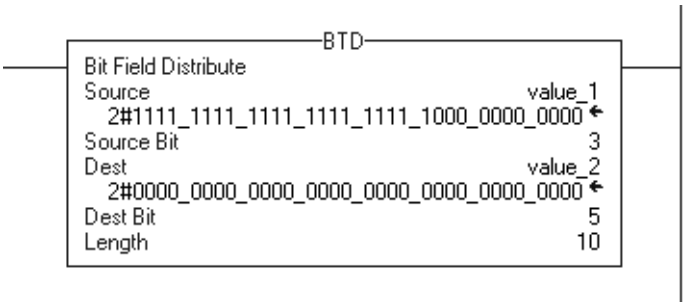


Una vez habilitada, la instrucción BTD mueve los bits dentro de *value_1*.

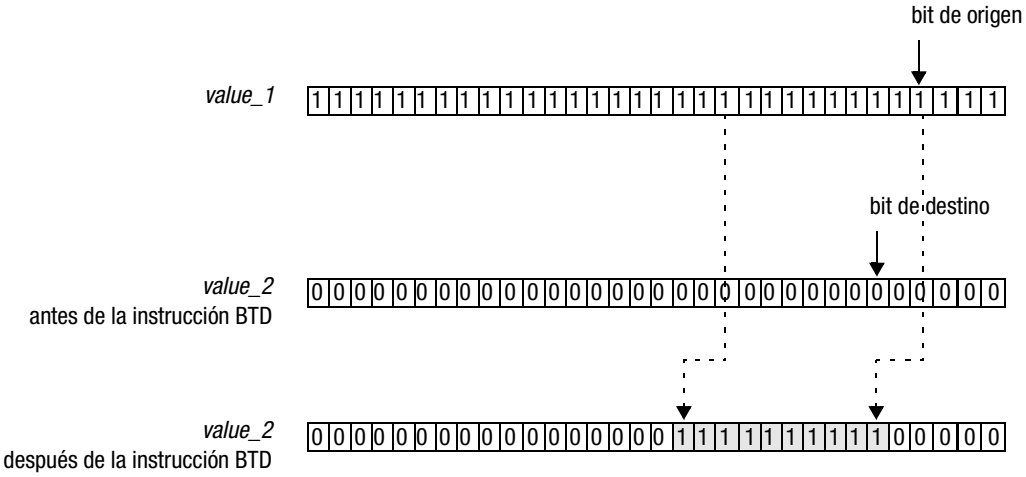

bit de destino
bit de origen
value_1
antes de la instrucción BTD
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
value_1
después de la instrucción BTD
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Las casillas sombreadas muestran los bits que se cambiaron en *value_1*.

ejemplo 2



Una vez habilitada, la instrucción BTD mueve 10 bits de *value_1* a *value_2*.



Las casillas sombreadas muestran los bits que se cambiaron en *value_2*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`BTD(source,source_bit,destination,destination_bit,length);`
texto ASCII	`BTD source source_bit destination destination_bit length`

Instrucciones relacionadas: CLR, MOV, MVM

Clear (CLR)

La instrucción CLR es una instrucción de salida.

Operandos:

Operando:	Tipo:	Formato:	Descripción:
Destino	SINT INT **DINT** **REAL**	tag	tag que se borra

Descripción: La instrucción CLR restablece todos los bits del destino.

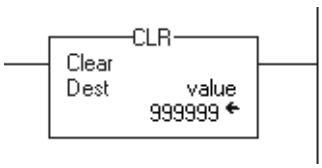
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción borra el destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de CLR:



Una vez habilitada, la instrucción CLR pone todos los bits de *value_1* a 0.

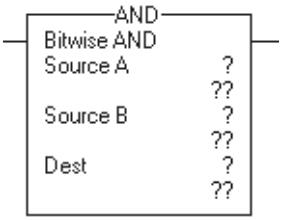
Otros formatos:

Formato:	Sintaxis:
texto neutro	`CLR(destination);`
texto ASCII	`CLR destination`

Instrucciones relacionadas: MOV

Bitwise AND (AND)

La instrucción AND es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT**	valor inmediato tag	valor en que se realiza la función Y con el origen B
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT**	valor inmediato tag	valor en que se realiza la función Y con el origen A
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT**	tag	almacena el resultado

Descripción: La instrucción AND realiza una función Y bit a bit usando los bits en los orígenes A y B y coloca el resultado en el destino.

Una vez habilitada, la instrucción evalúa la función Y:

Si el bit en el origen A es:	Y el bit en el origen B es:	El bit en el destino es:
0	0	0
0	1	0
1	0	0
1	1	1

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

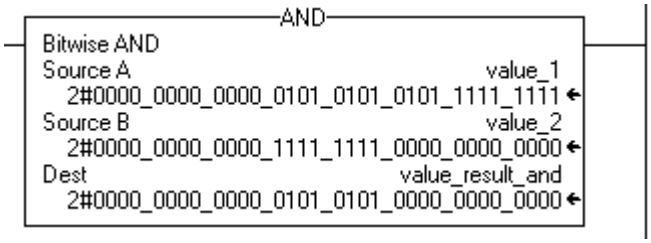
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción realiza una función Y bit a bit. La condición de salida de renglón está establecida como verdadera.

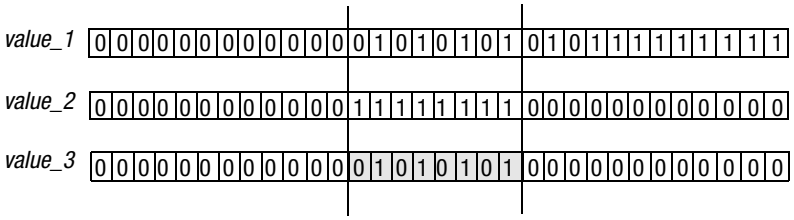
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de AND:



Cuando está habilitada, la instrucción AND realiza una función Y en *value_1* y *value_2* y coloca el resultado en *value_result_and*.



Las casillas sombreadas muestran los bits que se cambiaron.

Otros formatos:

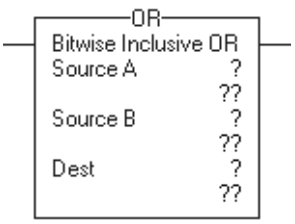
Formato:	Sintaxis:
texto neutro	`AND(source_A,source_B,destination);`
texto ASCII	`AND source_A source_B destination`

Instrucciones relacionadas: NOT, OR, XOR

Bitwise OR (OR)

La instrucción OR es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT **DINT**	valor inmediato tag	valor en que se realiza la función O con el origen B
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT **DINT**	valor inmediato tag	valor en que se realiza la función O con el origen A
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT**	tag	almacena el resultado

Descripción: La instrucción OR realiza una función O bit a bit usando los bits en los orígenes A y B y coloca el resultado en el destino.

Una vez habilitada, la instrucción evalúa la función O:

Si el bit en el origen A es:	Y el bit en el origen B es:	El bit en el destino es:
0	0	0
0	1	1
1	0	1
1	1	1

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

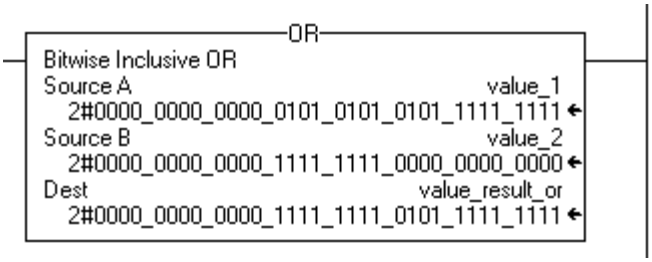
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción realiza una función O bit a bit. La condición de salida de renglón está establecida como verdadera.

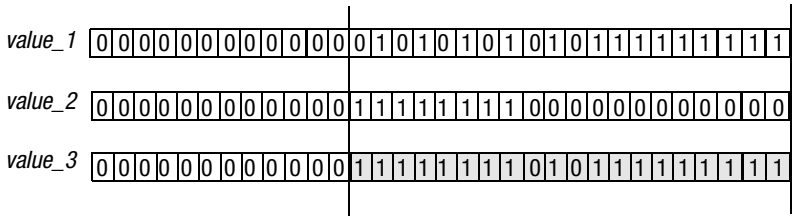
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de OR:



Cuando está habilitada, la instrucción OR realiza una función O en *value_1* y *value_2* y coloca el resultado en *value_result_or*.



Las casillas sombreadas muestran los bits que se cambiaron.

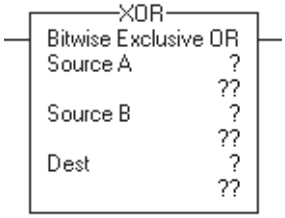
Otros formatos:

Formato:	Sintaxis:
texto neutro	`OR(source_A,source_B,destination);`
texto ASCII	`OR source_A source_B destination`

Instrucciones relacionadas: AND, OR, XOR

Bitwise Exclusive OR (XOR)

La instrucción XOR es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT DINT	valor inmediato tag	valor en que se realiza la función XOR con el origen B
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Origen B	SINT INT DINT	valor inmediato tag	valor en que se realiza la función XOR con el origen A
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT DINT	tag	almacena el resultado

Descripción: La instrucción XOR realiza una función O exclusivo bit a bit usando los bits en los orígenes A y B y coloca el resultado en el destino.

Una vez habilitada, la instrucción evalúa la función O exclusivo:

Si el bit en el origen A es:	Y el bit en el origen B es:	El bit en el destino es:
0	0	0
0	1	1
1	0	1
1	1	0

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

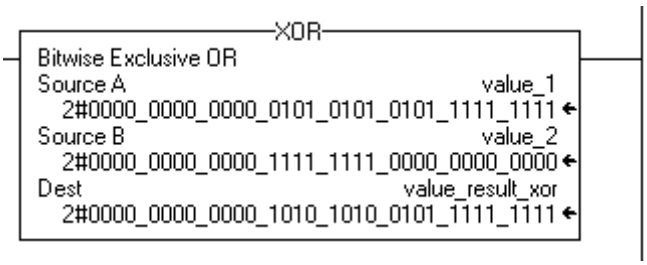
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción realiza una función O exclusivo bit a bit. La condición de salida de renglón se establece como verdadera.

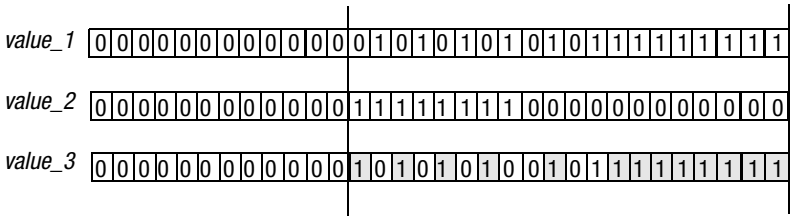
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de XOR:



Cuando está habilitada, la instrucción XOR realiza una función XOR en *value_1* y *value_2* y coloca el resultado en *value_result_or*.



Las casillas sombreadas muestran los bits que se cambiaron.

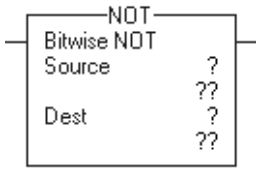
Otros formatos:

Formato:	Sintaxis:
texto neutro	`XOR(source_A,source_B,destination);`
texto ASCII	`XOR source_A source_B destination`

Instrucciones relacionadas: AND, NOT, OR

Bitwise NOT (NOT)

La instrucción NOT es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT	valor inmediato tag	valor en que se realiza la función NO
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT DINT	tag	almacena el resultado

Descripción: La instrucción NOT realiza una función NO bit a bit usando los bits en el origen y coloca el resultado en el destino.

Una vez habilitada, la instrucción evalúa la función NO:

Si el bit en el origen es:	El bit en el destino es:
0	1
1	0

Si combina los tipos de datos enteros, la instrucción llena con ceros los bits superiores de los tipos de datos enteros menores para que tengan el mismo tamaño que el tipo de datos más grande.

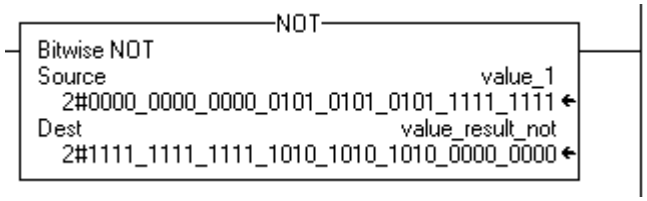
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La instrucción realiza una función NO bit a bit. La condición de salida de renglón está establecida como verdadera.

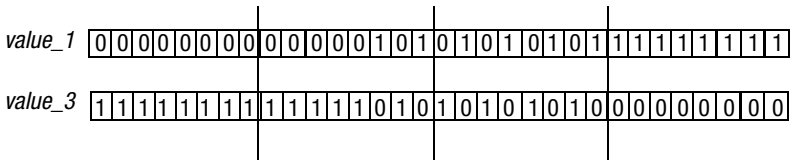
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de NOT:



Cuando está habilitada, la instrucción NOT realiza una función NOT en *value_1* y coloca el resultado en *value_result_not*.



Otros formatos:

Formato:	Sintaxis:
texto neutro	`NOT(source,destination);`
texto ASCII	`NOT source destination`

Instrucciones relacionadas: AND, OR, XOR

Instrucciones de arreglo (archivo)/ misceláneos

(FAL, FSC, COP, FLL, AVE, SRT, STD)

Introducción

Las instrucciones de archivo/varias realizan operaciones en arreglos de datos.

Si usted desea:	Use esta instrucción:	Vea la página:
realizar operaciones aritméticas, lógicas, de desplazamiento y función en valores en arreglos	FAL	7-7
buscar y comparar valores en arreglos	FSC	7-18
copiar el contenido de un arreglo a otro arreglo	COP	7-28
llenar un arreglo con datos específicos	FLL	7-32
calcular el promedio de un arreglo de valores	AVE	7-35
organizar una dimensión de datos de arreglo en orden ascendente	SRT	7-39
calcular la desviación estándar de un arreglo de valores	STD	7-42

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

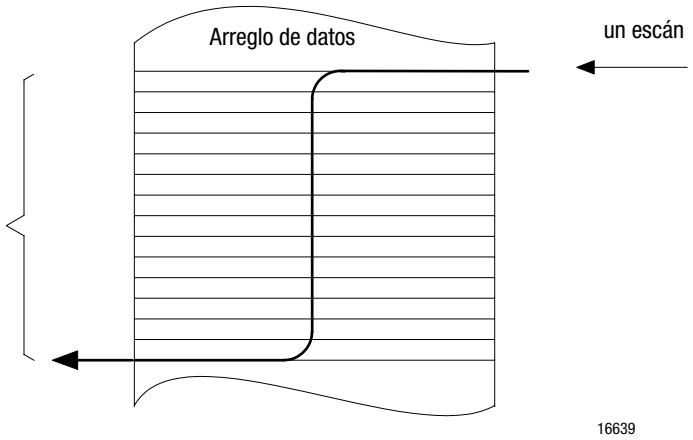
Cómo seleccionar el modo de operación

Para las instrucciones FAL y FSC, el modo indica al controlador cómo distribuir la operación del arreglo.

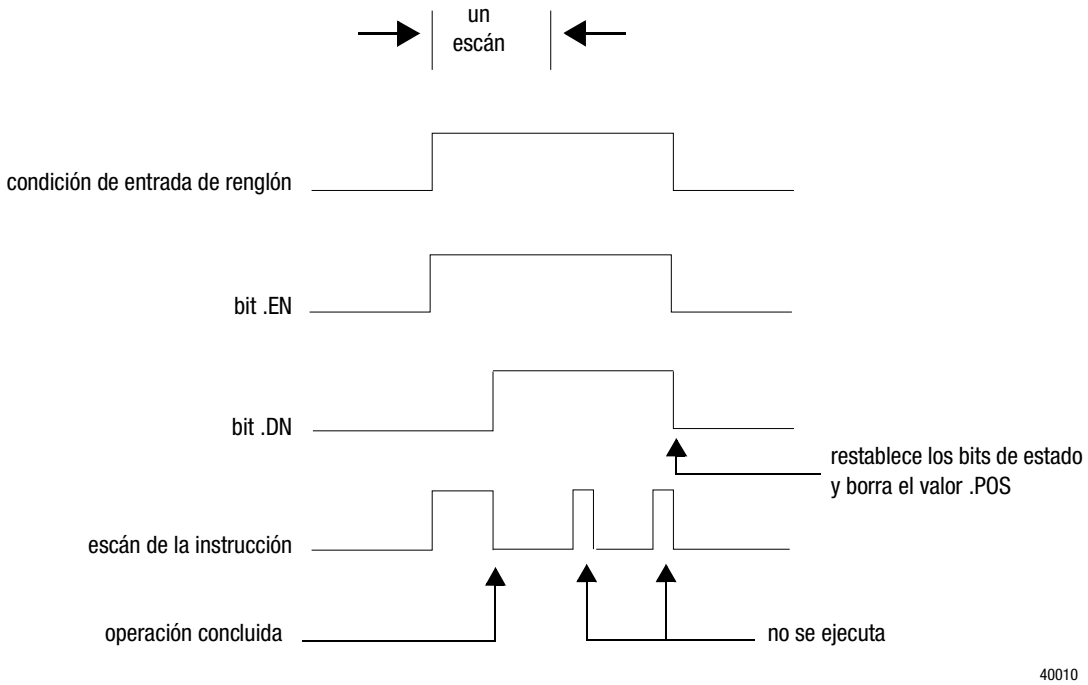
Si usted desea:	Seleccione este modo:
realizar una operación en todos los elementos especificados de un arreglo antes de continuar a la próxima instrucción	Todos
distribuir la operación de arreglo para un número de escanes introducir el número de elementos en que se realiza una operación por escán (1 – 2147483647)	Numérico
manipular un elemento del arreglo cada vez que la condición de entrada de renglón va de falso a verdadero	Incremental

Modo todos

En el modo todos, se realiza una operación en todos los elementos especificados en el arreglo antes de continuar a la próxima instrucción. La operación se inicia cuando la condición de entrada de renglón va de falso a verdadero. El valor de posición (.POS) en la estructura de control indica el elemento en el arreglo que la instrucción está usando actualmente. La operación se detiene cuando el valor .POS es igual al valor .LEN.



El siguiente diagrama de temporización indica la relación entre los bits de estado y la operación de la instrucción. Cuando se completa la ejecución de la instrucción, se establece el bit .DN. El bit .DN, el bit .EN y el valor .POS se borran cuando la condición de entrada de renglón es falsa. Sólo entonces se puede activar otra ejecución de la instrucción debido a una transición de falso a verdadero de una condición de entrada de renglón.

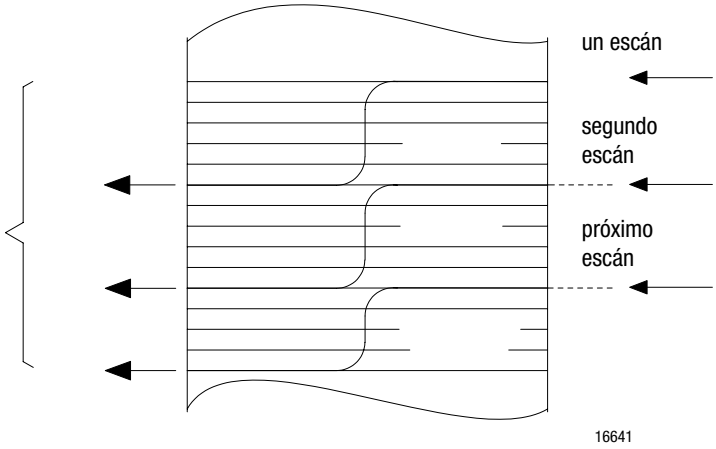


Modo numérico

El modo numérico distribuye la operación del arreglo para un número de escanes. Este modo es útil cuando se trabaja con datos sin tiempo crítico o grandes cantidades de datos. Usted introduce el número de elementos en que desea realizar una operación para cada escán, lo cual reduce el tiempo de escán.

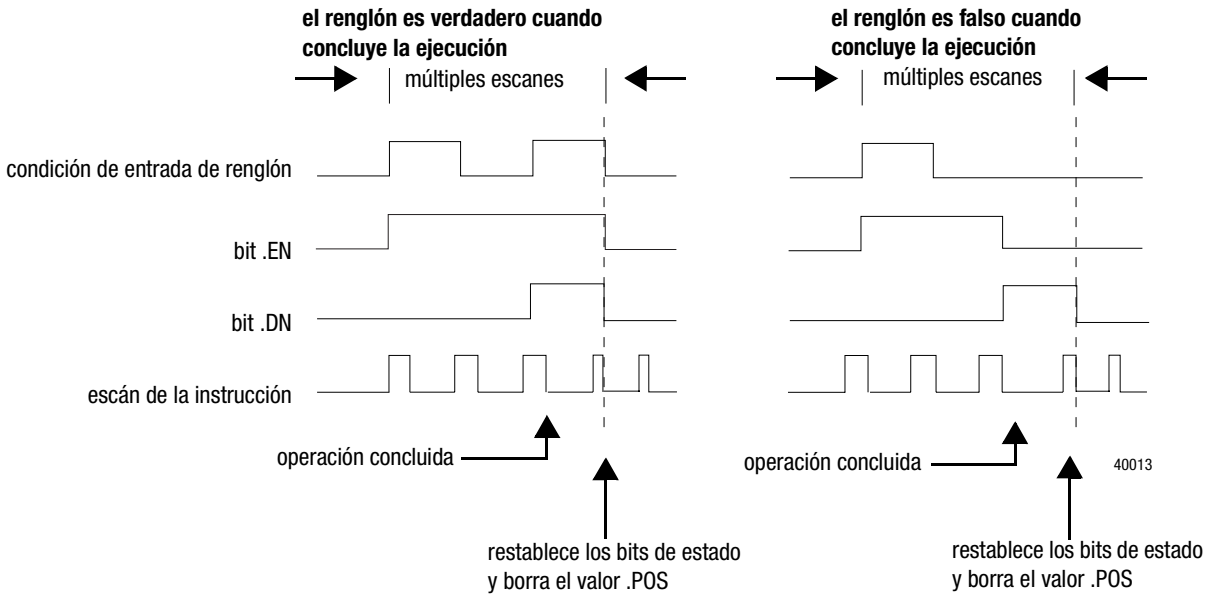
La ejecución se activa cuando la condición de entrada de renglón va de falso a verdadero. Una vez activada, la instrucción se ejecuta cada vez que se escanea durante el número de escanes necesarios para completar la operación en todo el arreglo. Una vez activada, la condición de entrada de

renglón se puede cambiar repetidamente sin interrumpir la ejecución de la instrucción.



Importante: Evite usar los resultados de una operación de instrucción de archivo en el modo numérico hasta que se establezca el bit .DN.

El siguiente diagrama de temporización indica la relación entre los bits de estado y la operación de la instrucción. Cuando se completa la ejecución de la instrucción, se establece el bit .DN.

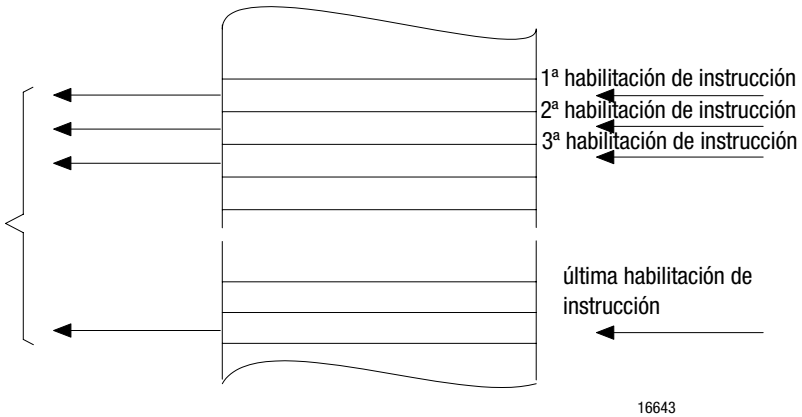

el renglón es verdadero cuando concluye la ejecución
múltiples escanes
el renglón es falso cuando concluye la ejecución
múltiples escanes
condición de entrada de renglón
bit .EN
bit .DN
escán de la instrucción
operación concluida
operación concluida
40013
restablece los bits de estado y borra el valor .POS
restablece los bits de estado y borra el valor .POS

Si la condición de entrada de renglón es verdadera al completarse la ejecución, los bits .EN y .DN se establecen hasta que la condición de entrada de renglón se hace falsa. Cuando la condición de entrada de renglón se hace falsa, estos bits se restablecen y el valor .POS se borra.

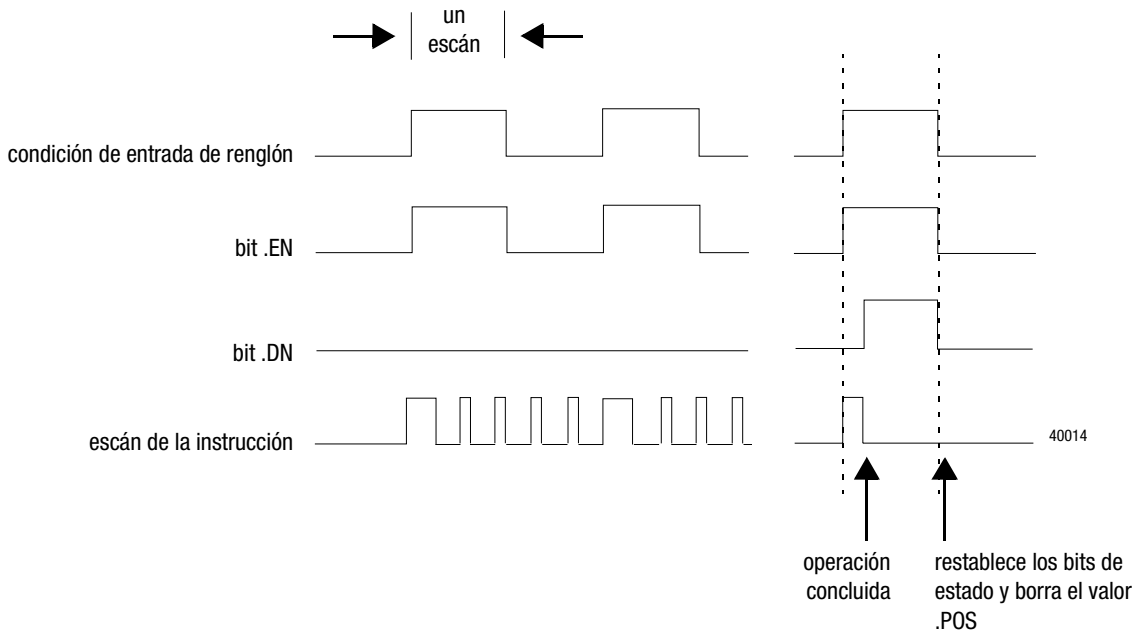
Si la condición de entrada de renglón es falsa al concluir la ejecución, el bit .EN se restablece inmediatamente. El bit .DN y el valor .POS se restablecen un escán después de que se restablece el bit .EN.

Modo incremental

El modo incremental manipula un elemento del arreglo cada vez que la condición de entrada de renglón de la instrucción va de falso a verdadero.



El siguiente diagrama de temporización indica la relación entre los bits de estado y la operación de instrucción. La ejecución ocurre solamente en un escán en el cual la condición de entrada de renglón va de falso a verdadero. Cada vez que ocurre esto, se manipula solamente un elemento del arreglo. Si la condición de entrada de renglón permanece verdadera durante más de un escán, la instrucción se ejecuta solamente durante el primer escán.



El bit .EN se establece cuando la condición de entrada de renglón es verdadera. El bit .DN se establece cuando el último elemento en el arreglo ha sido manipulado. Cuando el último elemento ha sido manipulado y la condición de entrada de renglón se hace falsa, el bit .EN, el bit .DN y el valor .POS se restablecen.

La diferencia entre el modo incremental y el modo numérico en un régimen de un elemento por escán es:

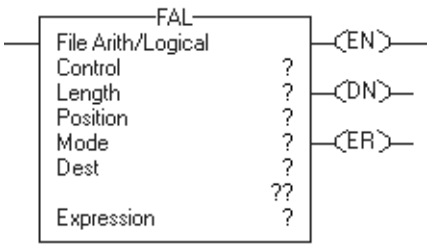
- El modo numérico con cualquier número de elementos por escán requiere solamente una transición de falso a verdadero de la condición de entrada de renglón para iniciar la ejecución. La instrucción continúa ejecutando el número especificado de elementos durante cada escán hasta concluir independientemente del estado de la condición de entrada de renglón.

- El modo incremental requiere que la condición de entrada de renglón cambie de falsa a verdadera para manipular un elemento en el arreglo.

File Arithmetic and Logic (FAL)

La instrucción FAL es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Control	CONTROL	tag	estructura de control para la operación
Longitud	DINT	valor inmediato	número de elementos en el arreglo que se manipulan
Posición	DINT	valor inmediato	elemento actual en el arreglo el valor inicial es típicamente 0
Modo	DINT	valor inmediato	cómo distribuir la operación seleccione INC, TODOS o introduzca un número
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado
Expresión	SINT INT **DINT** **REAL**	valor inmediato tag	una expresión que consiste en tags y/o valores inmediatos separados por operadores. Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.

Estructura CONTROL:

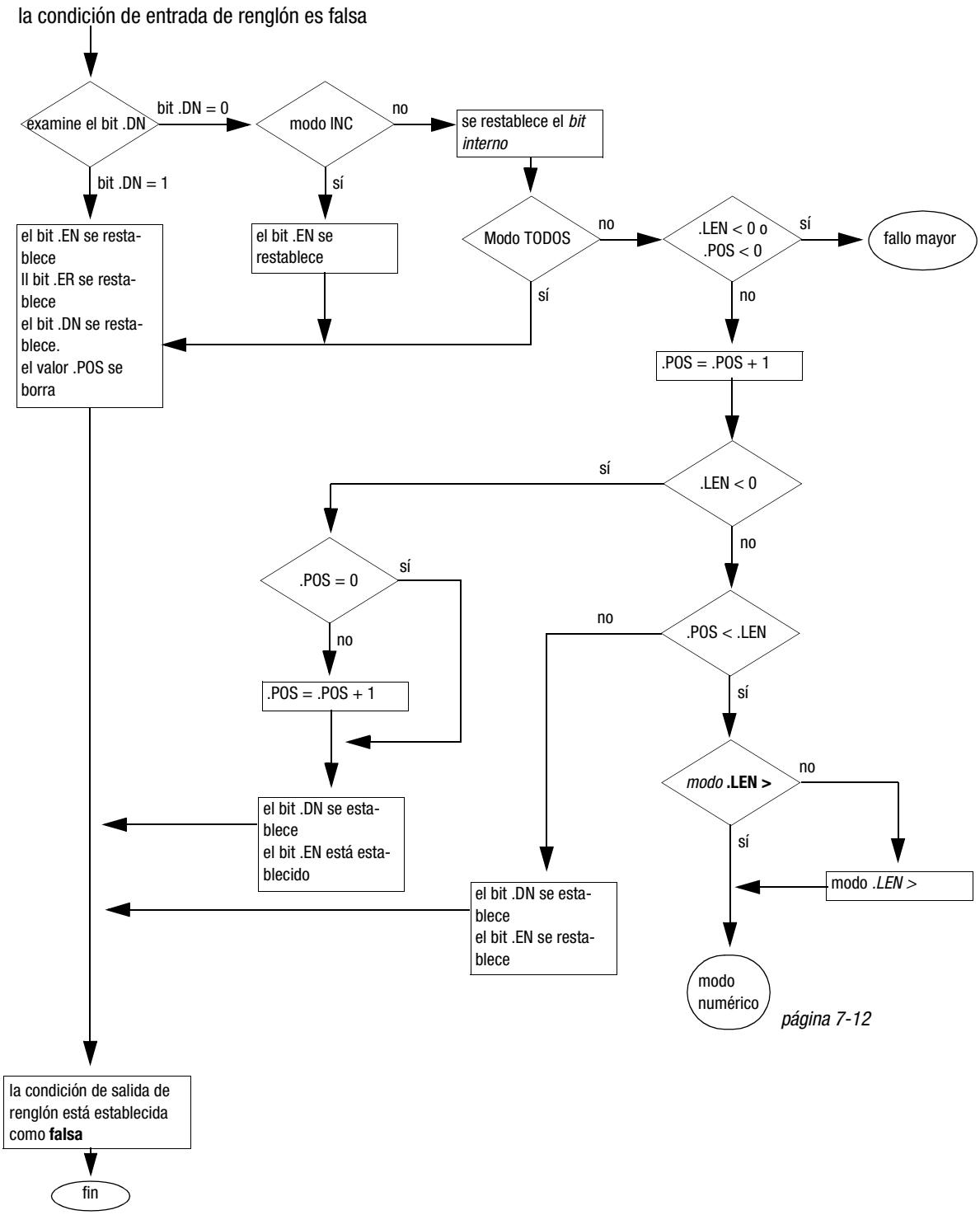
Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción FAL está habilitada.
.DN	BOOL	El bit de efectuado se establece cuando la instrucción ha efectuado la operación en el último elemento (.POS = .LEN).
.ER	BOOL	El bit de error se establece si la expresión genera un overflow (se establece S:V). La instrucción ya no se ejecuta hasta que el programa restablece el bit .ER. El valor .POS contiene la posición del elemento que causó el overflow.
.LEN	DINT	La longitud especifica el número de elementos en el arreglo en que la instrucción realiza una operación.
.POS	DINT	La posición contiene la posición del elemento actual al cual la instrucción obtiene acceso.

Descripción: La instrucción FAL realiza operaciones de copia, aritméticas, lógicas y función en los datos almacenados en un arreglo. La instrucción FAL realiza las mismas operaciones en los arreglos que la instrucción CPT realiza en los elementos.

El ejemplo que aparece en la página 7-13 muestra cómo usar el valor .POS para procesar el arreglo. Si un subíndice de la expresión del destino se encuentra fuera de rango, la instrucción FAL genera un fallo mayor (tipo 4, código 20).

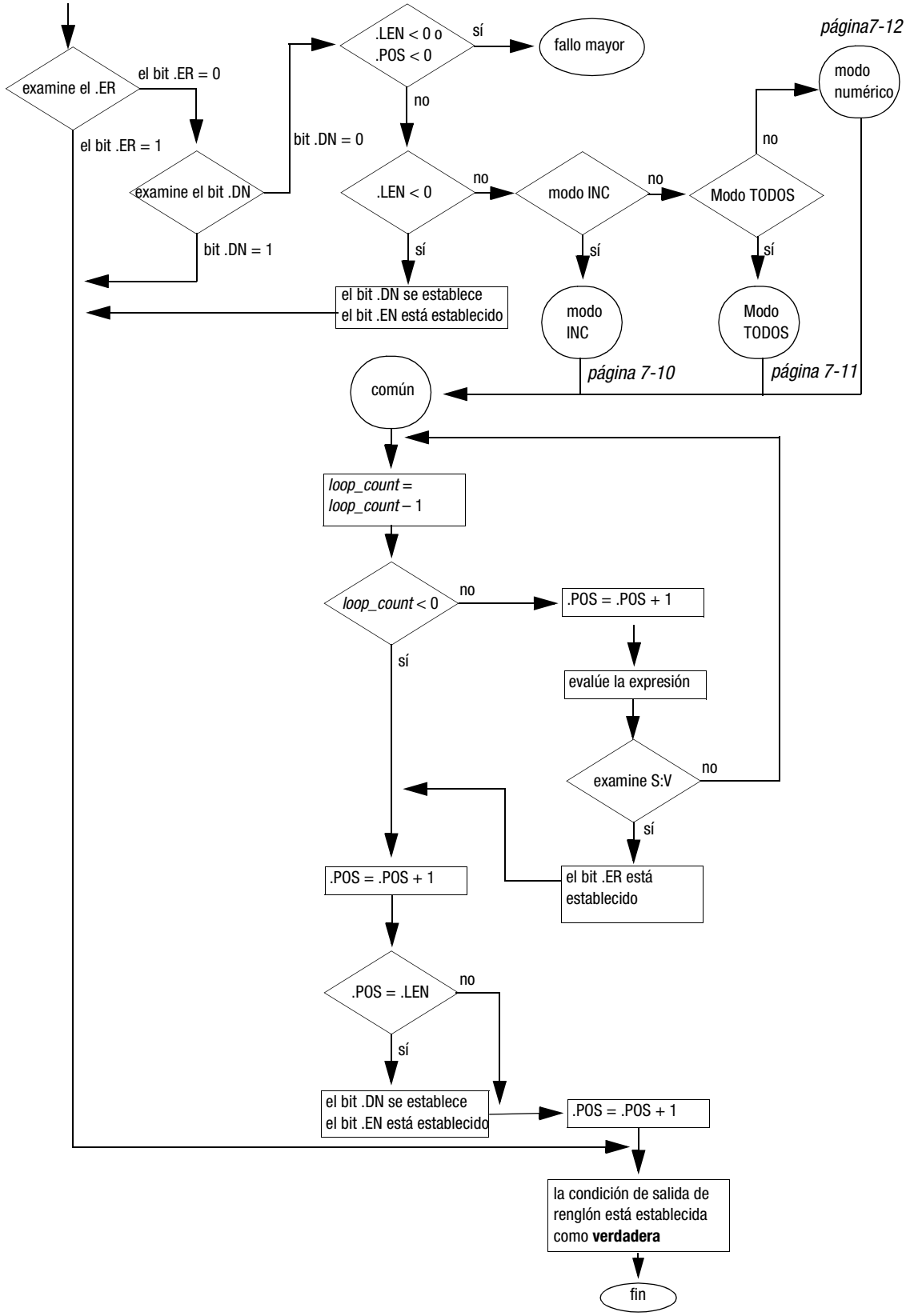
Ejecución

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.

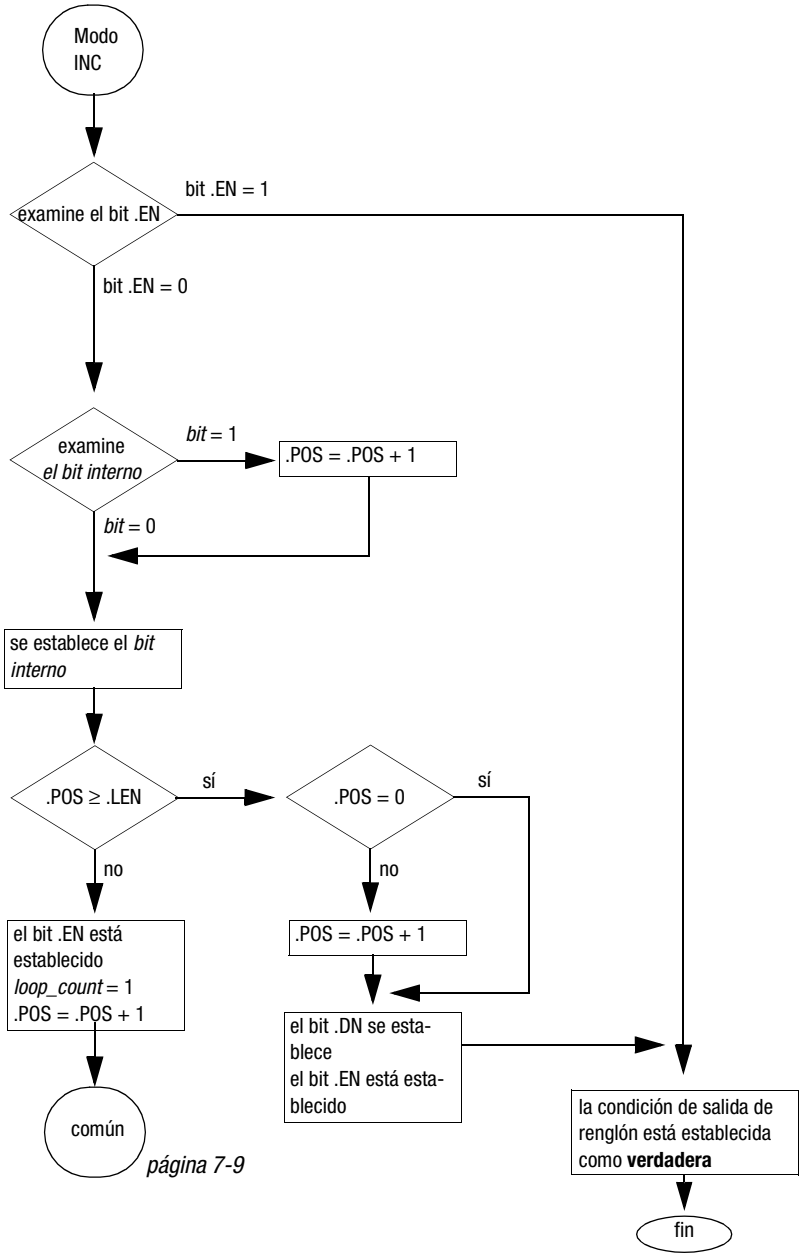


Condición: **Acción:**

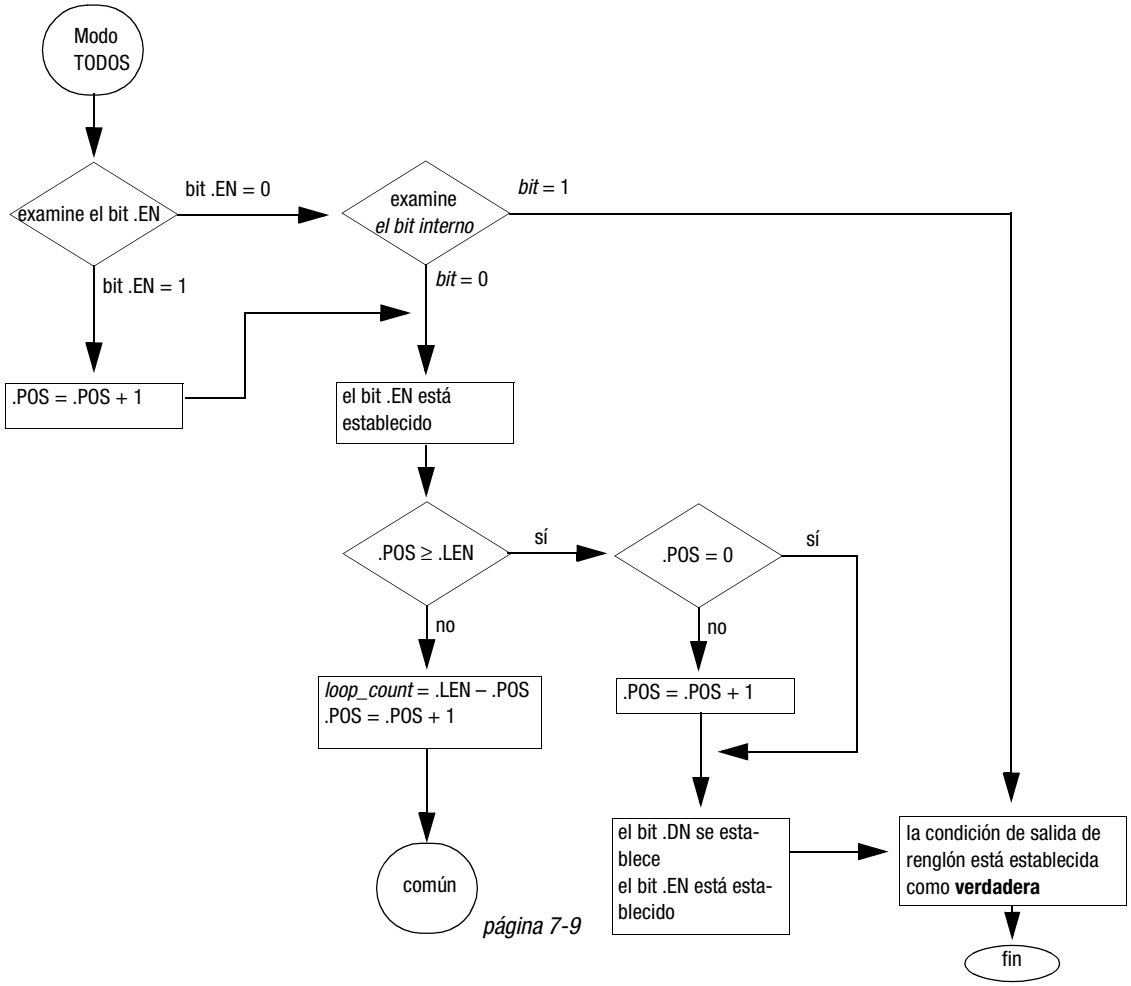
la condición de entrada de renglón es verdadera



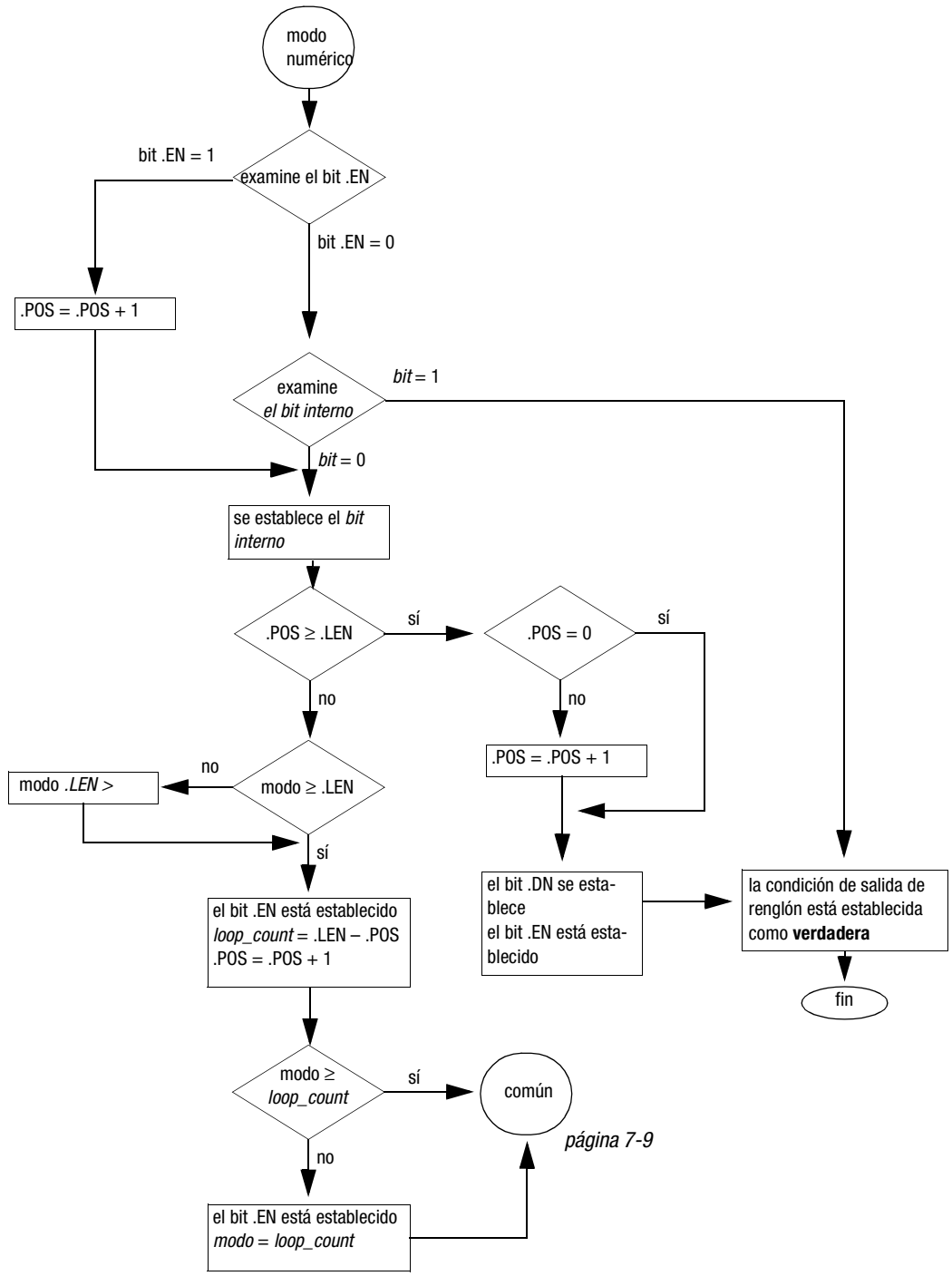
Condición: **Acción:**



Condición: **Acción:**



Condición: **Acción:**



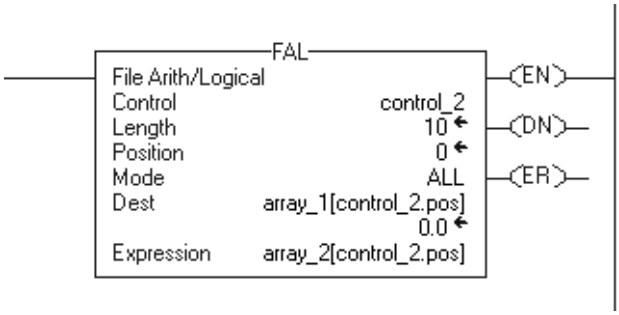
Indicadores de estado aritmético: Los indicadores de estado aritmético están afectados.

Condiciones de fallo:

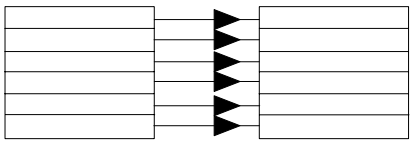
Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
el subíndice está fuera de rango	4	20
.POS < 0 ó .LEN < 0	4	21

Ejemplo de FAL:

copia de arreglo a arreglo



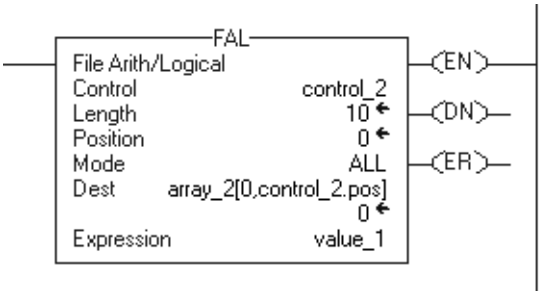
Una vez habilitada, la instrucción FAL copia cada elemento de *array_2* en la misma posición dentro de *array_1*.



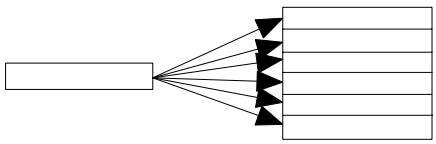
Expresión *array_2[control_2.pos]*

Destino *array_1[control_2.pos]*

copia de elemento a arreglo



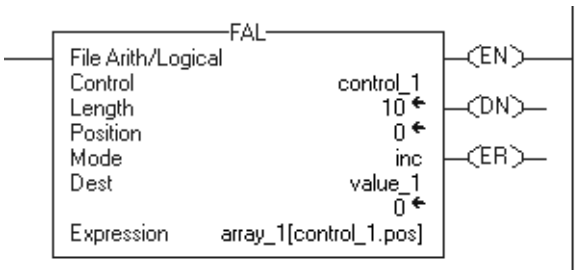
Una vez habilitada, la instrucción FAL copia *value_1* en las 10 primeras posiciones de la segunda dimensión de *array_2*.



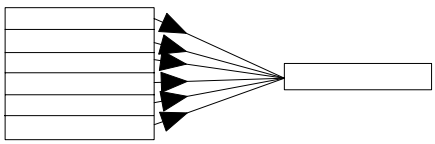
Expresión *value_1*

Destino *array_2[0,control_2.pos]*

copia de arreglo a elemento



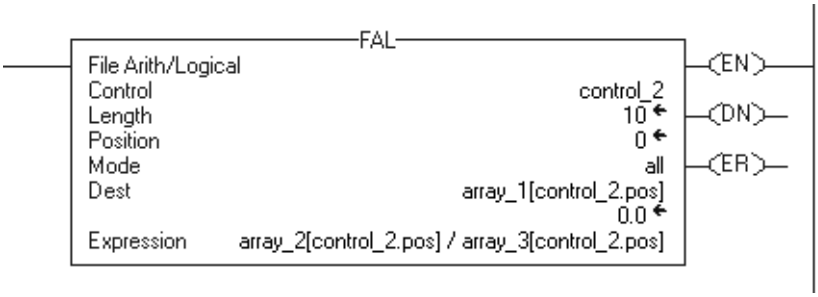
Cada vez que se habilita la instrucción FAL, ésta copia el valor actual de *array_1* a *value_1*. La instrucción FAL usa el modo incremental, por lo tanto se copia solamente un arreglo cada vez que se habilita la instrucción. La próxima vez que se habilita la instrucción, ésta sobrescribe *value_1* con el próximo valor en *array_1*.



Expresión *array_1[control_1.pos]*

Destino *value_1*

operación aritmética: (arreglo/arreglo) a arreglo

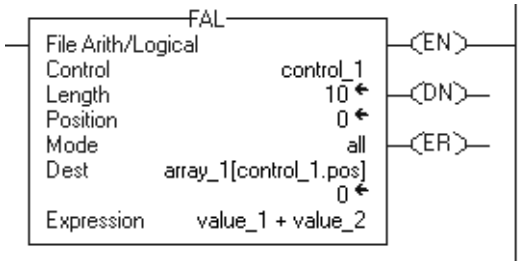


Una vez habilitada, la instrucción FAL divide el valor en la posición actual de *array_2* entre el valor en la posición actual de *array_3* y almacena el resultado en la posición actual de *array_1*.

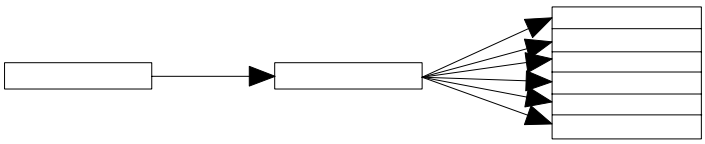
Expresión
array_2[control_2.pos] / array_3[control_2.pos]

Destino
array_1[control_2.pos]

operación aritmética: (elemento + elemento) a arreglo



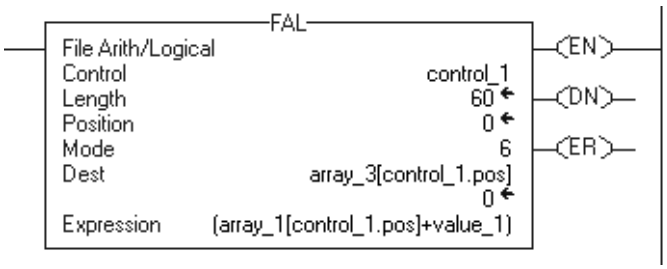
Una vez habilitada, la instrucción FAL suma *value_1* y *value_2* y almacena el resultado en la posición actual de *array_1*.



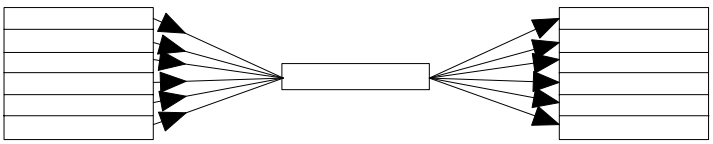
Expresión
value_1 + value_2

Destino
array_1[control_1.pos]

operación aritmética: (arreglo + elemento) a arreglo



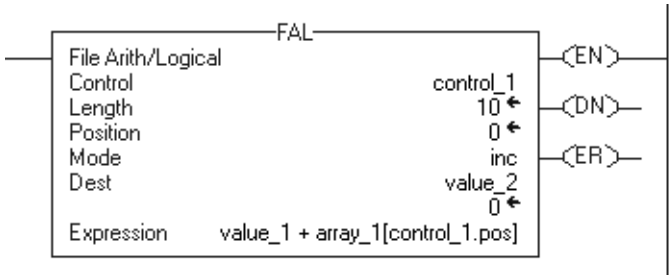
Una vez habilitada, la instrucción FAL suma el valor en la posición actual de *array_1* a *value_1* y almacena el resultado en la posición actual en *array_3*. La instrucción se debe ejecutar 10 veces para que *array_1* y *array_3* se manipulen por entero.



Expresión
array_1[control_1.pos] + value_1

Destino
array_3[control_1.pos]

operación aritmética: (elemento + arreglo) a elemento

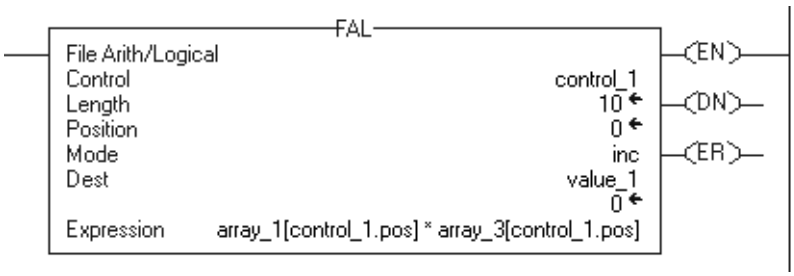


Cada vez que se habilita la instrucción FAL, ésta suma *value_1* al valor actual de *array_1* y almacena el resultado en *value_2*. La instrucción FAL usa el modo incremental, por lo tanto se suma solamente un valor de arreglo a *value_1* cada vez que se habilita la instrucción. La próxima vez que se habilita la instrucción, ésta sobrescribe *value_2*.

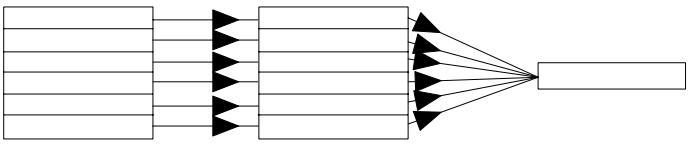
Expresión
value_1 + array_1[control_1.pos]

Destino
value_2

operación aritmética: (arreglo ∗ arreglo) a elemento



Una vez habilitada, la instrucción FAL multiplica el valor actual de *array_1* por el valor actual de *array_3* y almacena el resultado en *value_1*. La instrucción FAL usa el modo incremental, por lo tanto se multiplica solamente un par de valores de arreglo cada vez que se habilita la instrucción. La próxima vez que se habilita la instrucción, ésta sobrescribe *value_1*.



Expresión
*array_1[control_1.pos] * array_3[control_1.pos]*

Destino
value_1

Otros formatos:

Formato:	Sintaxis:
texto neutro	`FAL(control,length,position,mode,destination,expression);`
texto ASCII	`FAL control length position mode destination expression`

Instrucciones relacionadas: CPT, CMP, FSC

Usted programa las expresiones en las instrucciones FAL de la misma manera que las expresiones en las instrucciones CPT. Use las secciones siguientes para obtener información acerca de operadores válidos, formato y orden de operación, los cuales son comunes en ambas instrucciones.

Operadores válidos

Operador:	Descripción:	Optimo:
+	sumar	DINT, REAL
–	restar/cambiar signo	DINT, REAL
*	multiplicar	DINT, REAL
/	dividir	DINT, REAL
**	exponente (x a y)	DINT, REAL
ABS	valor absoluto	DINT, REAL
ACS	arco coseno	REAL
AND	función Y	DINT
ASN	arco seno	REAL
ATN	arco tangente	REAL
COS	coseno	REAL
DEG	radianes a grados	DINT, REAL
FRD	BCD a número entero	DINT

Operador:	Descripción:	Optimo:
LN	logaritmo natural	REAL
LOG	logaritmo de base 10	REAL
MOD	módulo-división	DINT, REAL
NOT	complemento bit a bit	DINT
OR	función O	DINT
RAD	radianes a grados	DINT, REAL
SIN	seno	REAL
SQR	raíz cuadrada	DINT, REAL
TAN	tangente	REAL
TOD	número entero a BCD	DINT
TRN	truncar	DINT, REAL
XOR	OR exclusivo, bit a bit	DINT

Cómo formatear expresiones

Para cada operador que usted usa en una expresión, tiene que proporcionar uno o dos operandos (tags o valores inmediatos). Use la tabla siguiente para formatear los operadores y operandos dentro de una expresión:

Para los operadores que realizan la operación en:	Use este formato:	Ejemplos:
un operando	operador (operando)	ABS(*tag_a*)
dos operandos	operand_a operador operand_b	• *tag_b* + 5 • *tag_c* AND *tag_d* • (*tag_e* ** 2) MOD (*tag_f* / *tag_g*)

Cómo determinar el orden de operación

Las operaciones que usted escribe en la expresión son realizadas por la instrucción en un orden determinado y no necesariamente según el orden en que las escribió. Puede anular el orden de operación agrupando los términos entre paréntesis, lo cual causa que la instrucción realice una operación entre paréntesis antes que las otras operaciones.

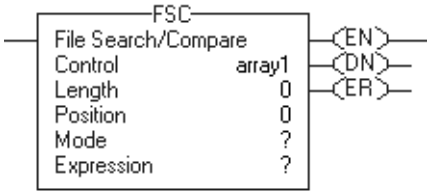
Las operaciones de orden igual se realizan desde la izquierda hacia la derecha.

Orden:	Operación:
1.	( )
2.	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3.	**
4.	– (cambiar signo), NOT
5.	*, /, MOD
6.	– (restar), +
7.	AND
8.	XOR
9.	OR

File Search and Compare (FSC)

La instrucción FCS es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
control	CONTROL	tag	estructura de control para la operación
Longitud	DINT	valor inmediato	número de elementos en el arreglo que se manipulan
Posición	DINT	valor inmediato	offset en el arreglo el valor inicial es típicamente 0
Modo	DINT	valor inmediato	cómo distribuir la operación seleccione INC, TODOS o introduzca un número
Expresión	SINT INT **DINT** **REAL**	valor inmediato tag	una expresión que consiste en tags y/o valores inmediatos separados por operadores.
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción FSC está habilitada.
.DN	BOOL	El bit de efectuado se establece cuando la instrucción ha efectuado la operación en el último elemento (.POS = .LEN).
.ER	BOOL	El bit de error no se modifica.
.IN	BOOL	El bit de inhibición indica que la instrucción FSC ha detectado una comparación verdadera. Usted debe restablecer este bit para poder continuar la operación de buscar.
.FD	BOOL	El bit de encontrado indica que la instrucción FSC ha detectado una comparación verdadera.
.LEN	DINT	La longitud especifica el número de elementos en el arreglo en que la instrucción realiza una operación.
.POS	DINT	La posición contiene la posición del elemento actual al cual la instrucción obtiene acceso.

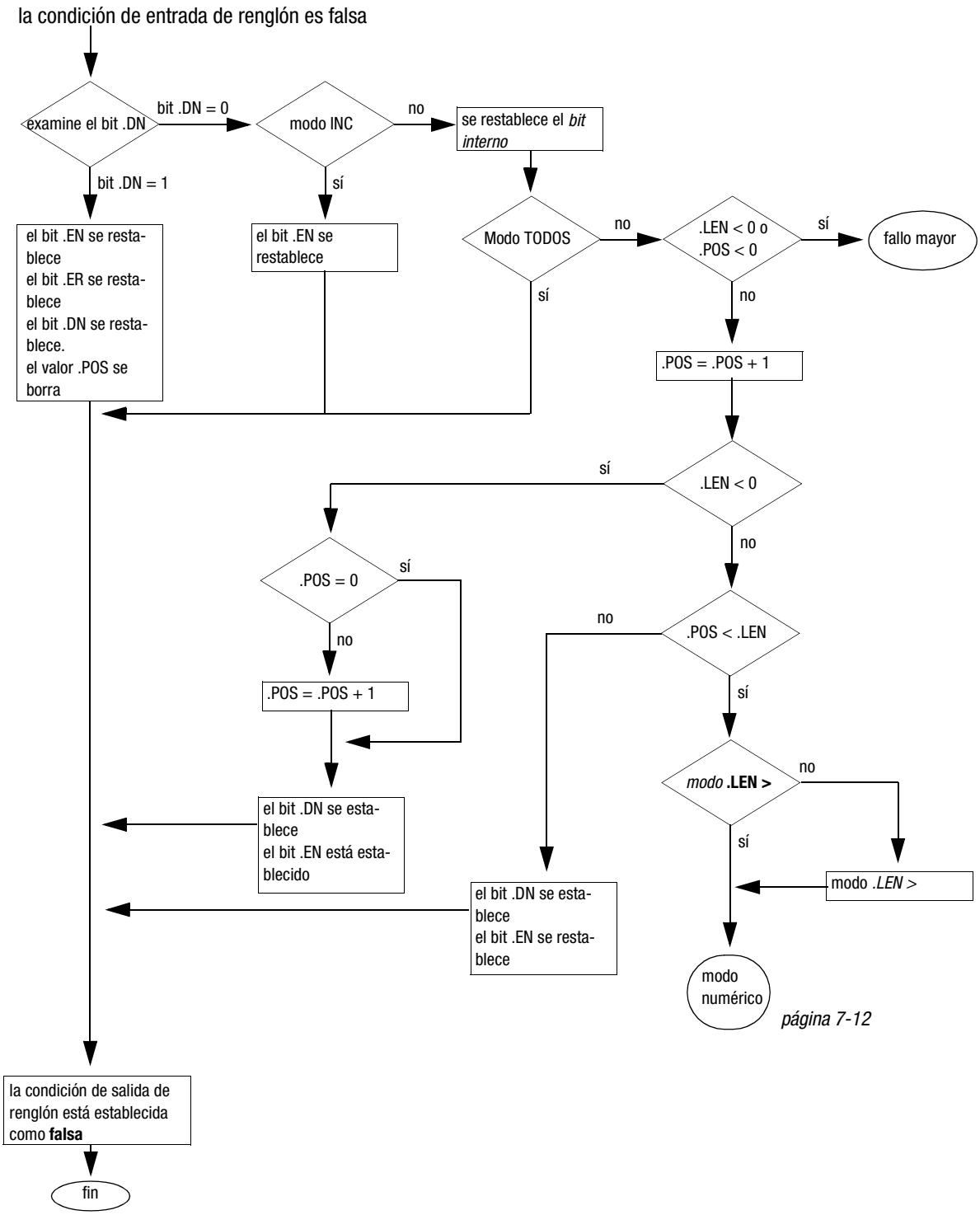
Descripción:

La instrucción FSC compara los valores en un arreglo, elemento por elemento, para las operaciones lógicas que se especifican en la expresión. Vea las sección *Cómo ver un arreglo como un conjunto de elementos* en la página B-1.

Cuando la instrucción FSC está habilitada y la comparación es verdadera, la instrucción establece el bit .FD y el bit .POS refleja la posición de arreglo donde la instrucción encontró la comparación verdadera. La instrucción establece el bit .IN para interrumpir la búsqueda.

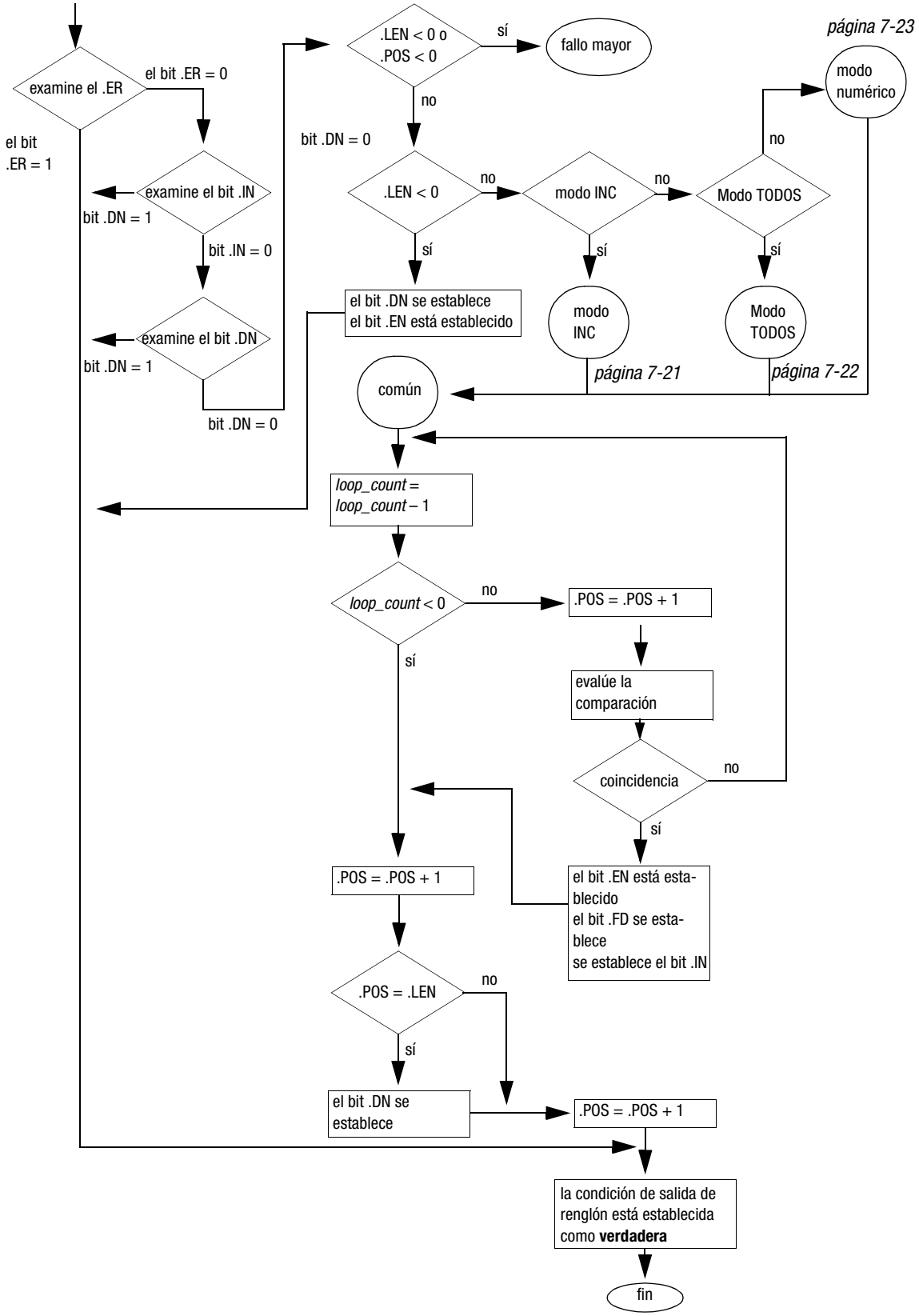
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.

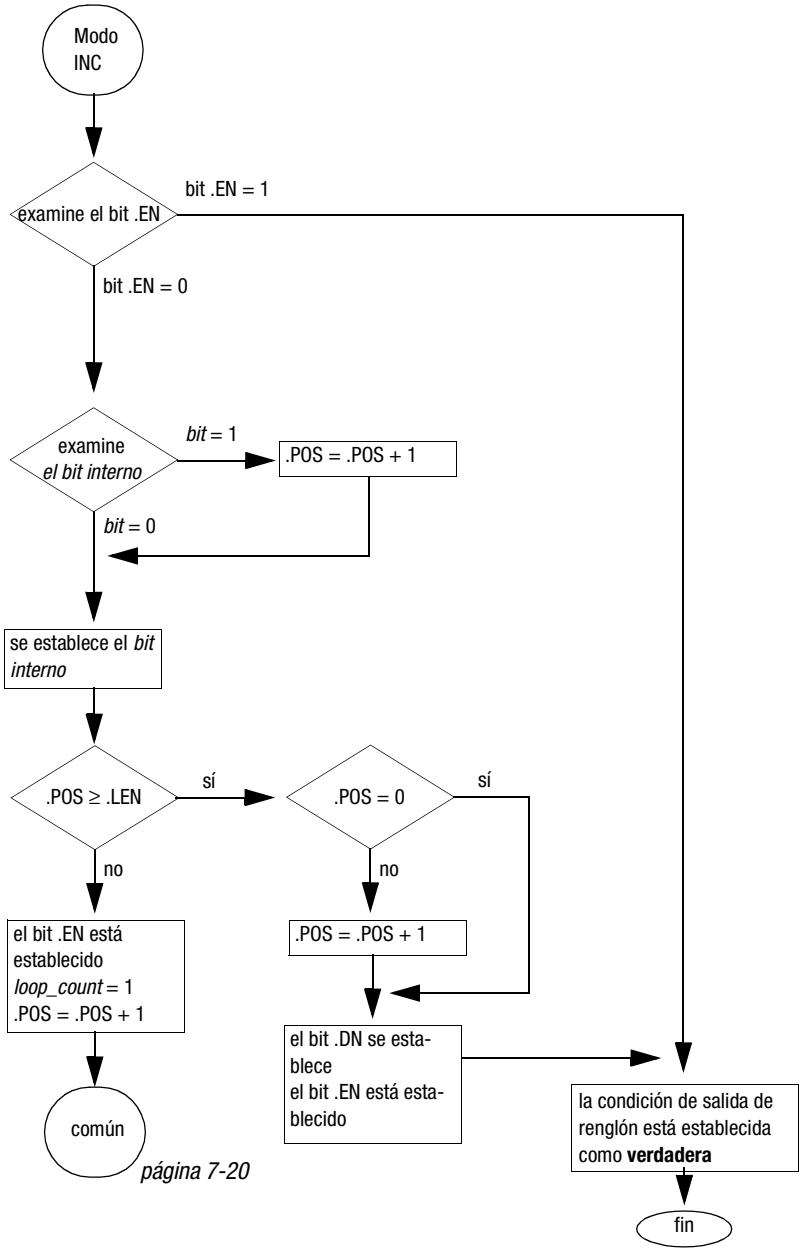


Condición: **Acción:**

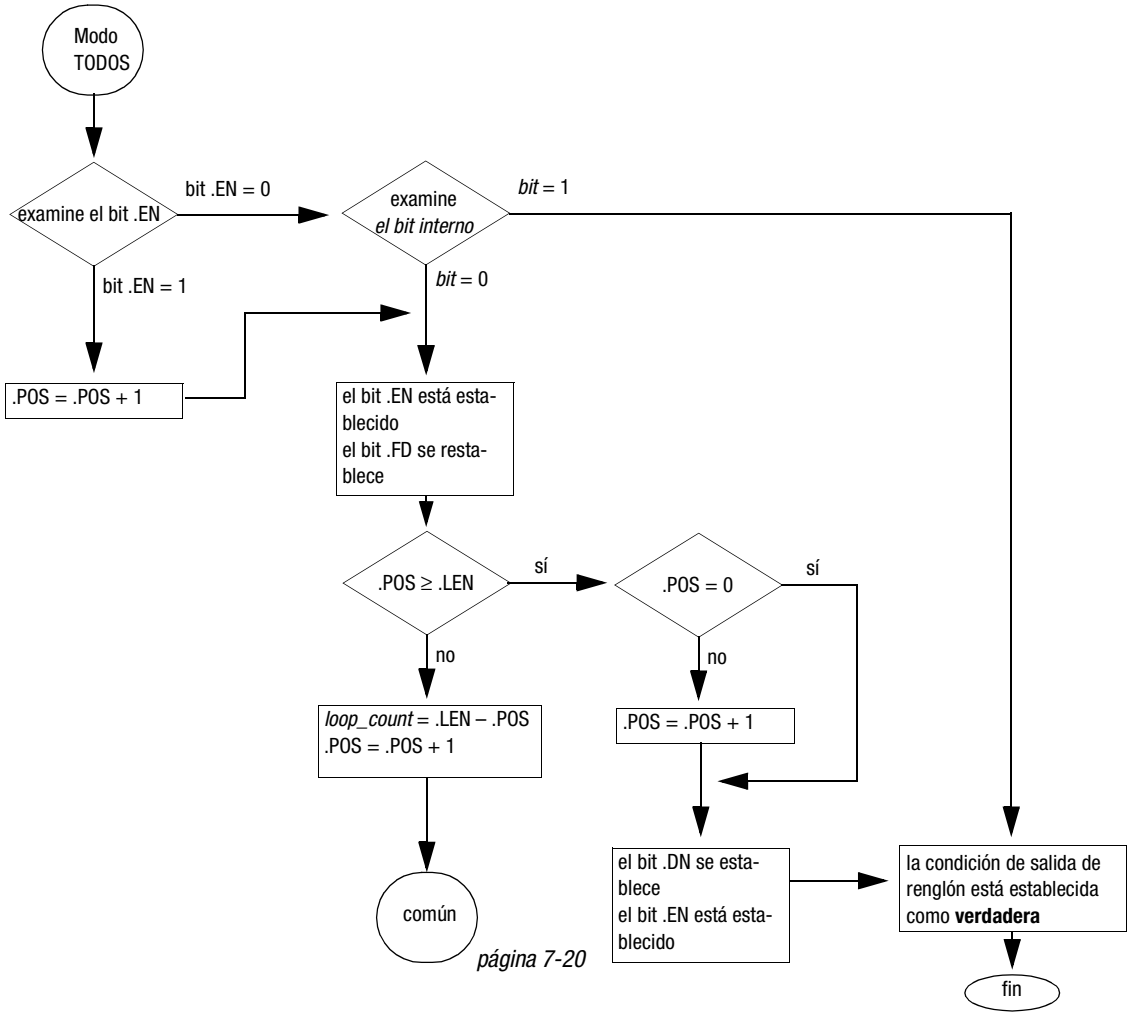
la condición de entrada de renglón es verdadera



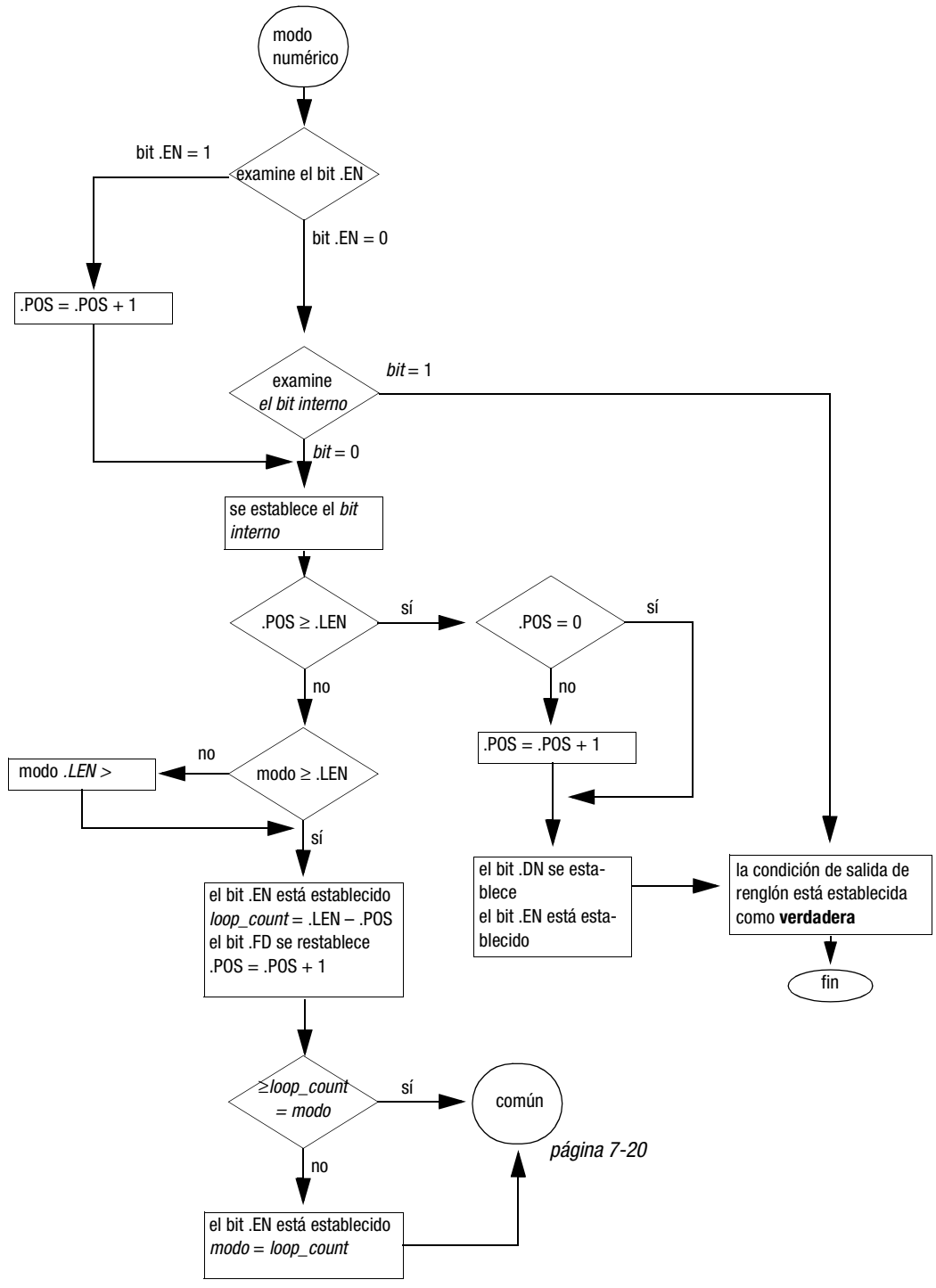
Condición: **Acción:**



Condición: **Acción:**



Condición: **Acción:**



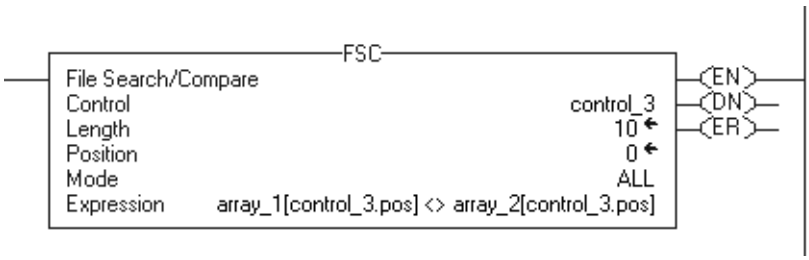
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.POS < 0 ó .LEN < 0	4	21

Ejemplo de FSC:

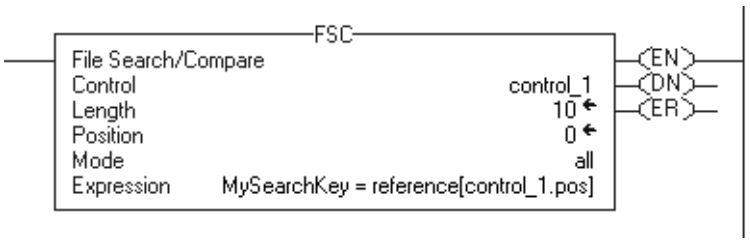
ejemplo 1
buscar una coincidencia entre dos arreglos



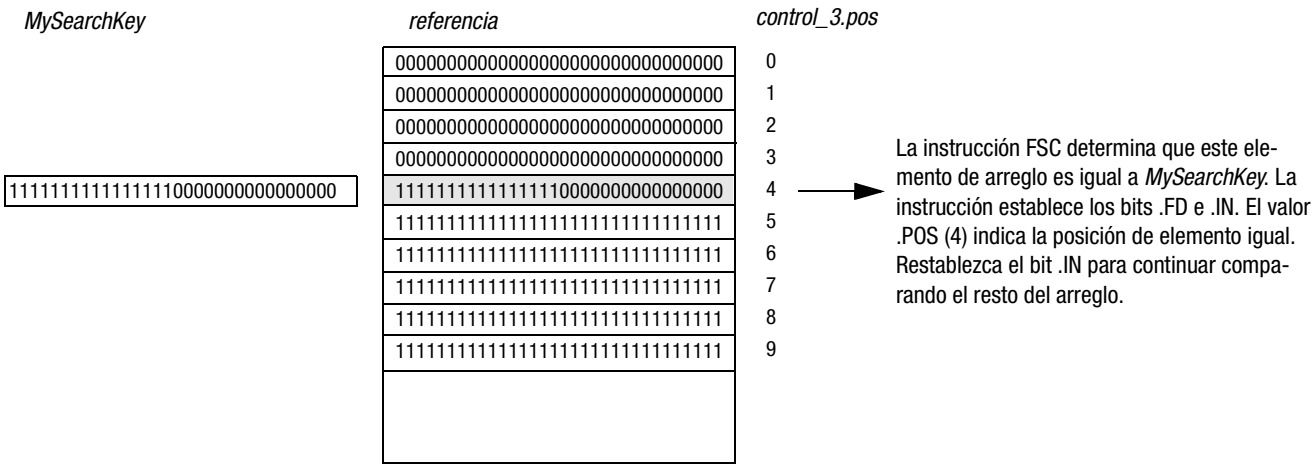
Una vez habilitada, la instrucción FSC compara cada uno de los 10 primeros elementos en *array_1* con los elementos correspondientes en *array_2*.

array_1	*array_2*	*control_3.pos*	
00000000000000000000000000000000	00000000000000000000000000000000	0	
00000000000000000000000000000000	00000000000000000000000000000000	1	
00000000000000000000000000000000	00000000000000000000000000000000	2	
00000000000000000000000000000000	00000000000000000000000000000000	3	
00000000000000001111111111111111	11111111111111110000000000000000	4	→ La instrucción FSC determina que estos elementos son diferentes. La instrucción establece los bits .FD e .IN. El valor .POS (4) indica la posición de los elementos que son diferentes. Restablezca el bit .IN para continuar comparando el resto del arreglo.
11111111111111111111111111111111	11111111111111111111111111111111	5	
11111111111111111111111111111111	11111111111111111111111111111111	6	
11111111111111111111111111111111	11111111111111111111111111111111	7	
11111111111111111111111111111111	11111111111111111111111111111111	8	
11111111111111111111111111111111	11111111111111111111111111111111	9	

ejemplo 2
buscar una coincidencia en un arreglo



Una vez habilitada, la instrucción FSC compara *MySearchKey* con los 10 elementos en *array_1*.



Otros formatos:

Formato:	Sintaxis:
texto neutro	`FSC(control,length,position,mode,expression);`
texto ASCII	`FSC control length position mode expression`

Instrucciones relacionadas: CMP, CPT, FAL

Usted programa las expresiones en las instrucciones FSC de la misma manera que las expresiones en las instrucciones CMP. Use las secciones siguientes para obtener información acerca de operadores válidos, formato y orden de operación, los cuales son comunes en ambas instrucciones.

Operadores válidos

Operador:	Descripción:	Optimo:
+	sumar	DINT, REAL
–	restar/cambiar signo	DINT, REAL
*	multiplicar	DINT, REAL
/	dividir	DINT, REAL
=	igual	DINT, REAL
<	menor que	DINT, REAL
<=	menor que o igual	DINT, REAL
>	mayor que	DINT, REAL
>=	mayor que o igual	DINT, REAL
<>	diferente	DINT, REAL
**	exponente (x a y)	DINT, REAL
ABS	valor absoluto	DINT, REAL
ACS	arco coseno	real
AND	función Y	DINT
ASN	arco seno	REAL
ATN	arco tangente	REAL

Operador:	Descripción:	Optimo:
COS	coseno	REAL
DEG	radianes a grados	DINT, REAL
FRD	BCD a número entero	DINT
LN	logaritmo natural	REAL
LOG	logaritmo de base 10	REAL
MOD	módulo-división	DINT, REAL
NOT	complemento bit a bit	DINT
OR	función O	DINT
RAD	radianes a grados	DINT, REAL
SIN	seno	REAL
SQR	raíz cuadrada	DINT, REAL
TAN	tangente	REAL
TOD	número entero a BCD	DINT
TRN	truncar	DINT, REAL
XOR	OR exclusivo, bit a bit	DINT

Cómo formatear expresiones

Para cada operador que usted usa en una expresión, tiene que proporcionar uno o dos operandos (tags o valores inmediatos). Use la tabla siguiente para formatear los operadores y operandos dentro de una expresión:

Para los operadores que realizan la operación en:	Use este formato:	Ejemplos:
un operando	operador (operando)	ABS(*tag_a*)
dos operandos	operand_a operador operand_b	• *tag_b* + 5 • *tag_c* AND *tag_d* • (*tag_e* ** 2) MOD (*tag_f* / *tag_g*)

Cómo determinar el orden de operación

Las operaciones que usted escribe en la expresión son realizadas por la instrucción en un orden determinado y no necesariamente según el orden en que las escribió. Puede anular el orden de operación agrupando los términos entre paréntesis, lo cual causa que la instrucción realice una operación entre paréntesis antes que las otras operaciones.

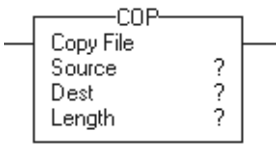
Las operaciones de orden igual se realizan desde la izquierda hacia la derecha.

Orden:	Operación:
1.	( )
2.	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3.	**
4.	– (cambiar signo), NOT
5.	*, /, MOD
6.	<, <=, >, >=, =
7.	– (restar), +
8.	AND
9.	XOR
10.	OR

File Copy (COP)

La instrucción COP es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	**SINT** **INT** **DINT** **REAL** estructura	tag	elemento inicial que se copia **Importante:** Los operandos de origen y destino deben ser del mismo tipo, de lo contrario pueden ocurrir resultados inesperados
Destino	**SINT** **INT** **DINT** **REAL** estructura	tag	el elemento inicial que va a ser sobrescrito por el origen **Importante:** Los operandos de origen y destino deben ser del mismo tipo, de lo contrario pueden ocurrir resultados inesperados
Longitud	DINT	valor inmediato tag	número de elementos de destino que se copian

Descripción: La instrucción COP copia los valores del origen al destino. El origen no se cambia. El número de bytes copiados es:

Conteo de bytes = longitud * (número de bytes en el tipo de datos de destino)

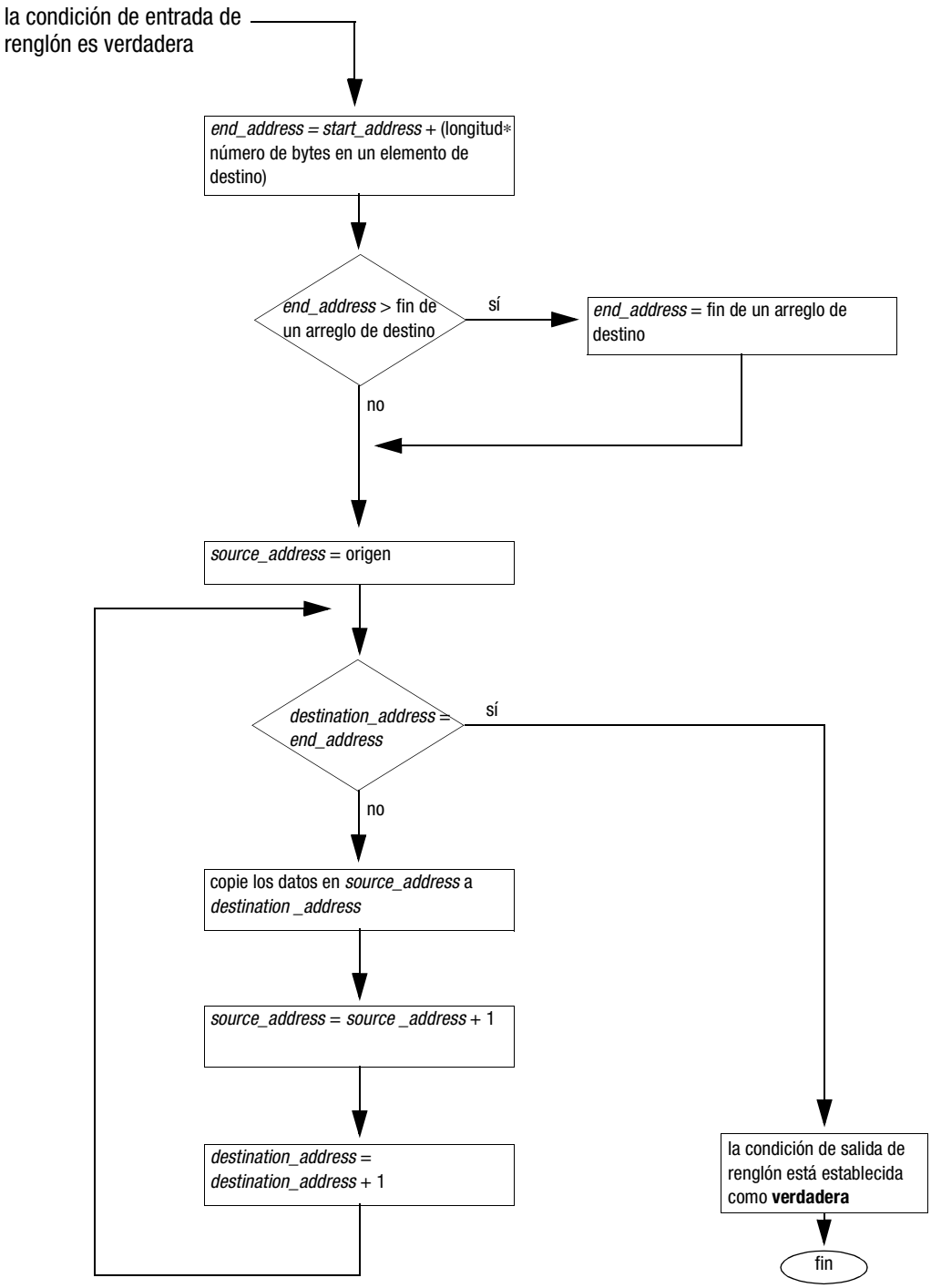
ATENCION: Si el conteo de bytes es mayor que la longitud del origen, se copian datos inciertos para el resto de los elementos.

La instrucción COP realiza la operación en la memoria de datos contiguos y realiza una copia directa de memoria de byte a byte, lo cual requiere entender la configuración de memoria del controlador. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información.

La instrucción COP no escribe más allá del fin del arreglo. Si la longitud es mayor que el número total de elementos en el arreglo de destino, la instrucción COP se detiene al final del arreglo. No se genera un fallo mayor.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.

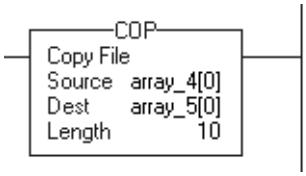

la condición de entrada de renglón es verdadera
end_address = start_address + (longitud* número de bytes en un elemento de destino)
end_address > fin de un arreglo de destino
sí
end_address = fin de un arreglo de destino
no
source_address = origen
destination_address = end_address
sí
no
copie los datos en source_address a destination _address
source_address = source _address + 1
destination_address = destination_address + 1
la condición de salida de renglón está establecida como verdadera
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

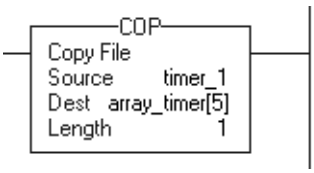
Ejemplo de COP:

ejemplo 1



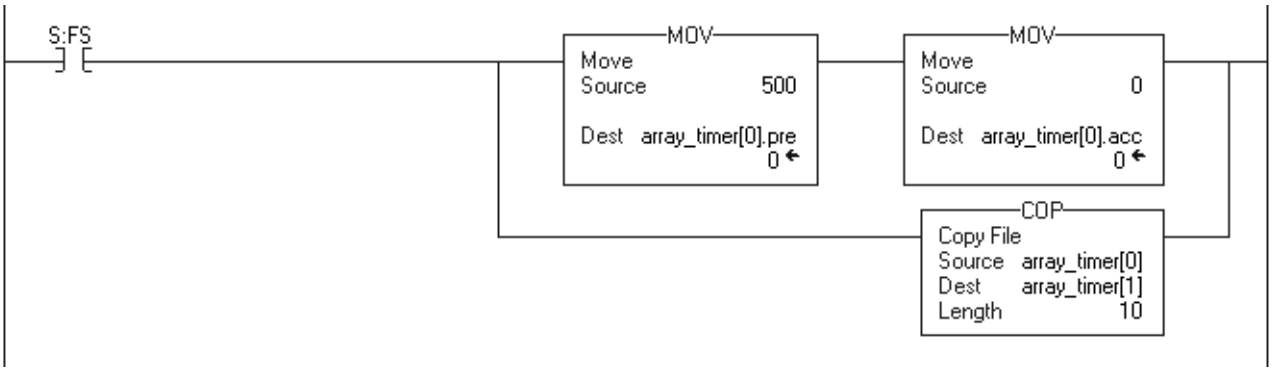
El *array_4* y *array_5* son del mismo tipo de datos. Una vez habilitada, la instrucción COP copia los 10 primeros elementos de *array_4* en los 10 primeros elementos de *array_5*.

ejemplo 2

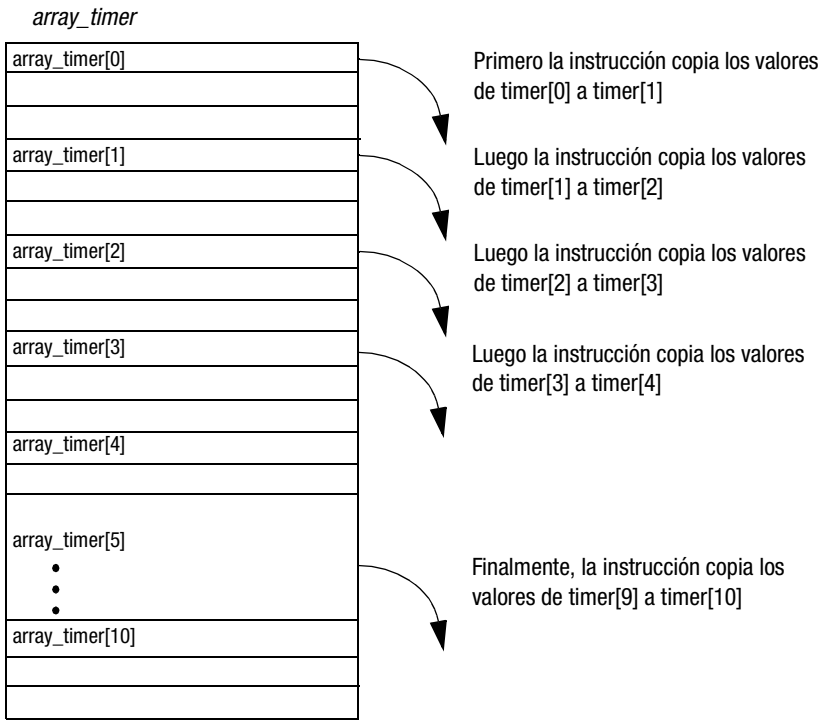


Una vez habilitada, la instrucción COP copia la estructura *timer_1* en el elemento 5 de *array_timer*. La instrucción copia solamente una estructura a un elemento de arreglo.

ejemplo 3



Este ejemplo inicializa un arreglo de estructuras de temporizador. Una vez habilitadas, las instrucciones MOV inicializan los valores .PRE y .ACC del primer elemento de *array_timer*. Una vez habilitada, la instrucción COP copia un bloque contiguo de bytes a partir de *array_timer[0]*. La longitud es nueve estructuras de temporizador.


array_timer
array_timer[0]
array_timer[1]
array_timer[2]
array_timer[3]
array_timer[4]
array_timer[5]
array_timer[10]
Primero la instrucción copia los valores de timer[0] a timer[1]
Luego la instrucción copia los valores de timer[1] a timer[2]
Luego la instrucción copia los valores de timer[2] a timer[3]
Luego la instrucción copia los valores de timer[3] a timer[4]
Finalmente, la instrucción copia los valores de timer[9] a timer[10]

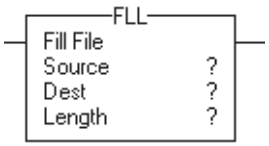
Otros formatos:

Formato:	Sintaxis:
texto neutro	`COP(source,destination,length);`
texto ASCII	`COP(source,destination,length)`

Instrucciones relacionadas: FAL, FLL, MOV

File Fill (FLL)

La instrucción FLL es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	**SINT** **INT** **DINT** **REAL**	valor inmediato tag	elemento que se copia **Importante:** Los operandos de origen y destino deben ser del mismo tipo, de lo contrario pueden ocurrir resultados inesperados
Destino	**SINT** **INT** **DINT** **REAL** estructura	tag	el elemento inicial que será sobrescrito por el origen **Importante:** Los operandos de origen y destino deben ser del mismo tipo, de lo contrario pueden ocurrir resultados inesperados El método preferido de inicializar una estructura es usar la instrucción COP.
Longitud	DINT	valor inmediato	número de elementos que se llenan

Descripción:

La instrucción FLL llena los elementos de un arreglo con el valor de origen. El origen no se cambia. El número de bytes llenados es:

Conteo de bytes = longitud ∗ (número de bytes en el tipo de datos de destino)

La instrucción FLL realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

La instrucción FLL no escribe más allá del fin de un arreglo. Si la longitud es mayor que el número total de elementos en el arreglo de destino, la instrucción FLL se detiene al final del arreglo. No se genera un fallo mayor.

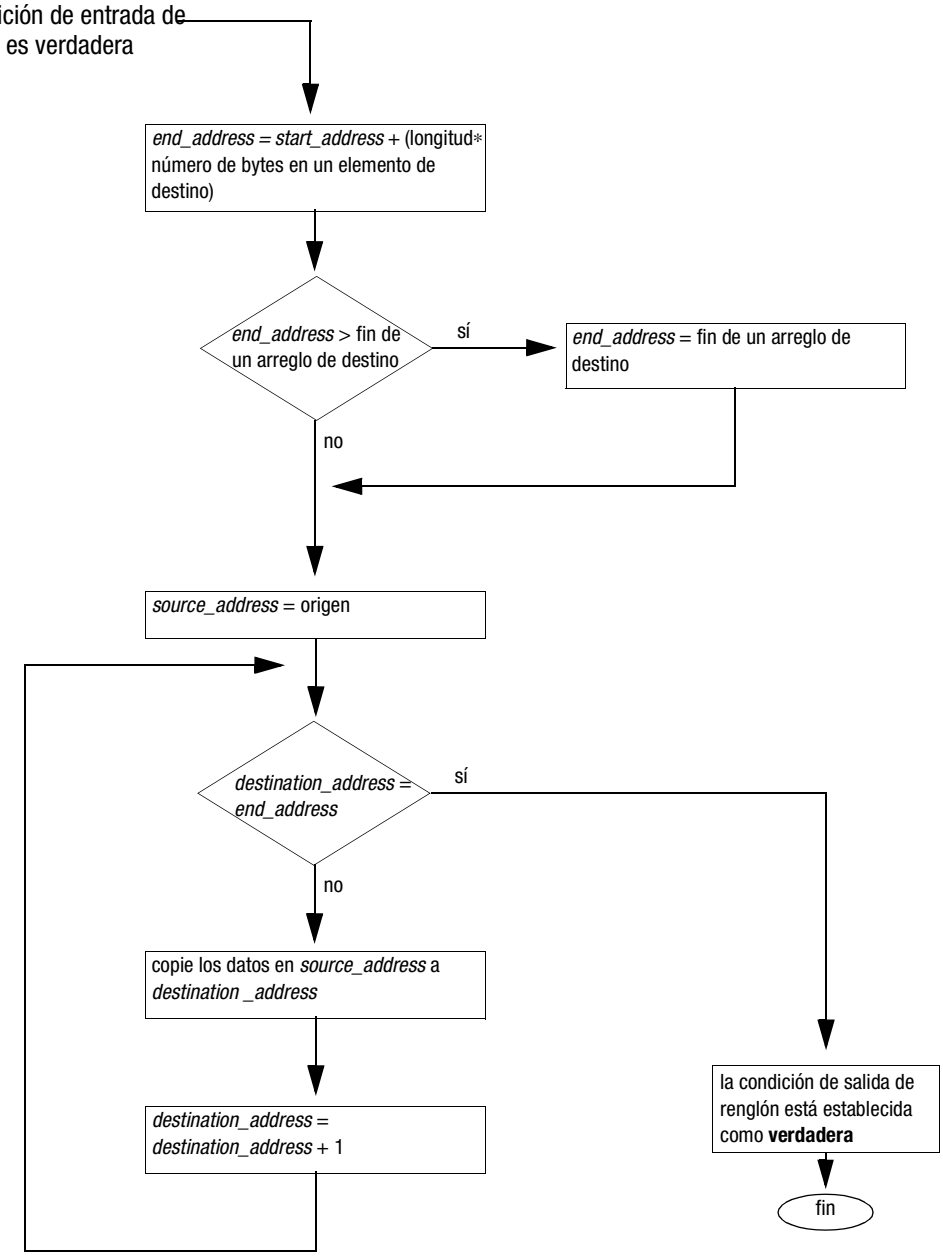
El origen y el destino deben ser del mismo tipo para obtener los resultados óptimos. Si usted desea llenar una estructura, use la instrucción COP (vea el ejemplo 3 en la página 7-31).Si combina los tipos de datos para el origen y destino, los elementos de destino se llenan con los valores de origen convertidos.

Si el origen es:	Y el destino es:	El origen se convierte en:
SINT, INT, DINT o REAL	SINT	SINT
SINT, INT, DINT o REAL	INT	INT
SINT, INT, DINT o REAL	DINT	DINT
SINT, INT, DINT o REAL	REAL	REAL
SINT	estructura	SINT (no se convierte)
INT	estructura	INT (no se convierte)
DINT	estructura	DINT (no se convierte)
REAL	estructura	REAL (no se convierte)

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.

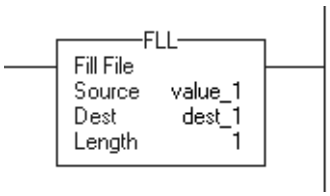
la condición de entrada de renglón es verdadera


end_address = start_address + (longitud* número de bytes en un elemento de destino)
end_address > fin de un arreglo de destino
sí
end_address = fin de un arreglo de destino
no
source_address = origen
destination_address = end_address
sí
no
copie los datos en source_address a destination _address
destination_address = destination_address + 1
la condición de salida de renglón está establecida como verdadera
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de FLL:



Una vez habilitada, la instrucción FLL copia el valor en *value_1* a *dest_1*.

Tipo de datos de origen (*value_1*):	Valor de origen (*value_1*):	Tipo de datos de destino (*dest_1*):	Valor de destino (*dest_1*) después de FLL:
SINT	16#80 (–128)	DINT	16#FFFF FF80 (–128)
DINT	16#1234 5678	SINT	16#78
SINT	16#01	REAL	1.0
REAL	2.0	INT	16#0002
SINT	16#01	TIMER	16#0101 0101 16#0101 0101 16#0101 0101
INT	16#0001	TIMER	16#0001 0001 16#0001 0001 16#0001 0001
DINT	16#0000 0001	TIMER	16#0000 0001 16#0000 0001 16#0000 0001

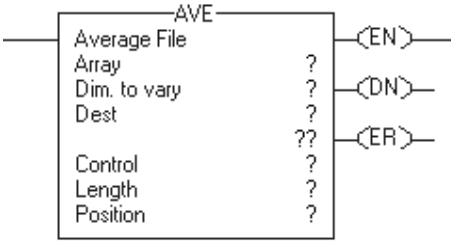
Otros formatos:

Formato:	Sintaxis:
texto neutro	`FLL(source,destination,length);`
texto ASCII	`FLL(source,destination,length)`

Instrucciones relacionadas: FAL, COP, MOV

File Average (AVE)

La instrucción AVE es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Arreglo	SINT INT DINT **REAL**	tag de arreglo	hallar el promedio de los valores en este arreglo especificar el primer elemento del grupo de elementos que se va a promediar **no** use CONTROL.POS en el subíndice
Dimensión para variar:	DINT	valor inmediato (0, 1, 2)	qué dimensión usar según el número de dimensiones, el orden es: array[dim_0,dim_1,dim_2] array[dim_0,dim_1] array[dim_0]
Destino	SINT INT DINT **REAL**	tag	resultado de la operación
Control	CONTROL	tag	estructura de control para la operación
Longitud	DINT	valor inmediato	número de elementos en el arreglo que se va a promediar
Posición	DINT	valor inmediato	elemento actual en el arreglo el valor inicial es típicamente 0

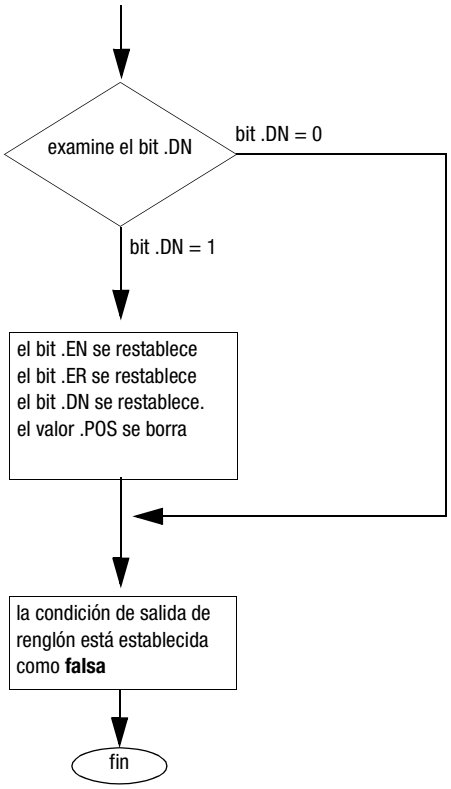
Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción AVE está habilitada.
.DN	BOOL	El bit de efectuado se establece cuando la instrucción ha realizado una operación en el último elemento del arreglo (.POS = .LEN).
.ER	BOOL	Se establece el bit de error si la instrucción genera un overflow. La instrucción ya no se ejecuta hasta que el programa restablece el bit .ER. La posición del elemento que causó el overflow se almacena en el valor .POS.
.LEN	DINT	La longitud especifica el número de elementos en el arreglo en que la instrucción realiza una operación.
.POS	DINT	La posición contiene la posición del elemento actual al cual la instrucción obtiene acceso.

Descripción: La instrucción AVE calcula el promedio de un conjunto de valores.

Importante: Asegúrese de que la longitud no cause que la instrucción exceda la dimensión especificada para variar. Si ocurre esto, el destino será incorrecto. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información.

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	
la condición de entrada de renglón es verdadera	La instrucción AVE calcula el promedio sumando todos los elementos especificados en el arreglo y dividiéndolos por el número de elementos. Internamente, la instrucción usa una instrucción FAL para calcular el promedio. Expresión = cálculo de promedio Modo = TODOS Para obtener detalles acerca de cómo se ejecuta la instrucción FAL, vea la página 7-8.

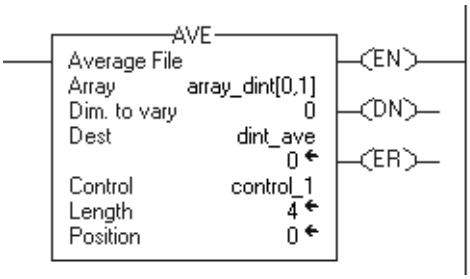
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.POS < 0 ó .LEN < 0	4	21
La dimensión que se va a variar no existe para el arreglo especificado	4	20

Ejemplo de AVE:

ejemplo 1



array_dint es DINT[4,5]

dimensión 1

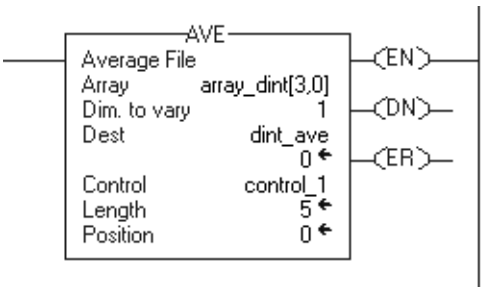
subíndices	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

dimensión 0

$$AVE = \frac{19 + 14 + 9 + 4}{4} = \frac{46}{4} = 11.5$$

$dint\_ave = 12$

ejemplo 2



array_dint es DINT[4,5]

dimensión 1

subíndices	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

dimensión 0

$$AVE = \frac{5 + 4 + 3 + 2 + 1}{5} = \frac{15}{5} = 3$$

$dint\_ave = 3$

Otros formatos:

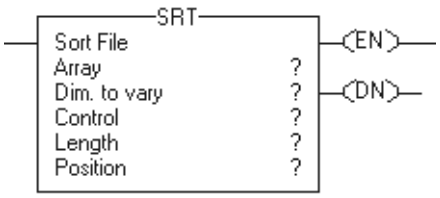
Formato:	Sintaxis:
texto neutro	`AVE(array,dim_to_vary,destination,control,length,position);`
texto ASCII	`AVE array dim_to_vary destination control length position`

Instrucciones relacionadas: SRT, STD

File Sort (SRT)

La instrucción SRT es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Arreglo	**SINT** **INT** **DINT** **REAL**	tag de arreglo	arreglo que se clasifica especificar el primer elemento del grupo de elementos que se clasifica **no** use CONTROL.POS en el subíndice
Dimensión para variar:	DINT	valor inmediato (0, 1, 2)	qué dimensión usar según el número de dimensiones, el orden es: array[dim_0,dim_1,dim_2] array[dim_0,dim_1] array[dim_0]
Control	CONTROL	tag	estructura de control para la operación
Longitud	DINT	valor inmediato	número de elementos en el arreglo que se clasifica
Posición	DINT	valor inmediato	elemento actual en el arreglo el valor inicial es típicamente 0

Estructura CONTROL:

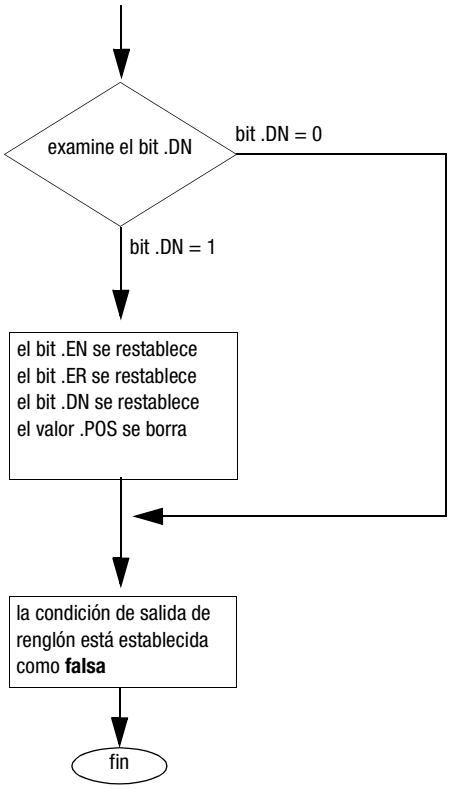
Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción SRT está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando los elementos especificados se han clasificado.
.ER	BOOL	Se establece el bit de error cuando .LEN < 0 ó .POS < 0. Cualquiera de estas condiciones también genera un fallo mayor.
.LEN	DINT	La longitud especifica el número de elementos en el arreglo en que la instrucción realiza una operación.
.POS	DINT	La posición contiene la posición del elemento actual al cual la instrucción obtiene acceso.

Descripción:

La instrucción SRT clasifica un conjunto de valores en una dimensión (dimensión para variar) del arreglo en orden ascendente.

Importante: Asegúrese de que la longitud no cause que la instrucción exceda la dimensión especificada para variar. Si ocurre esto, ocurrirán resultados inesperados. Vea la página, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información B-4.

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	
la condición de entrada de renglón es verdadera	La instrucción SRT clasifica los elementos especificados del arreglo en orden ascendente.

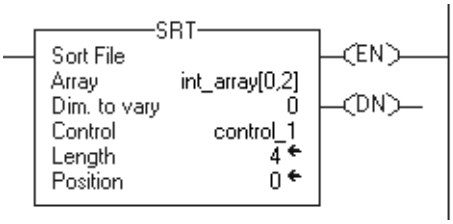
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.POS < 0 ó .LEN < 0	4	21
La dimensión que se va a variar no existe para el arreglo especificado	4	20
La instrucción intenta obtener acceso a los datos fuera de los límites del arreglo.	4	20

Ejemplo de SRT:

ejemplo 1



int _array es DINT[4,5]

Antes

dimensión 1

subíndices	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

dimensión 0

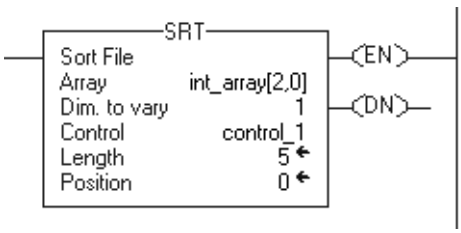
Después

dimensión 1

subíndices	0	1	2	3	4
0	20	19	3	17	16
1	15	14	8	12	11
2	10	9	13	7	6
3	5	4	18	2	1

dimensión 0

ejemplo 2



int _array es DINT[4,5]

Antes

dimensión 1

subíndices	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

dimensión 0

Después

dimensión 1

subíndices	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	6	7	8	9	10
3	5	4	3	2	1

dimensión 0

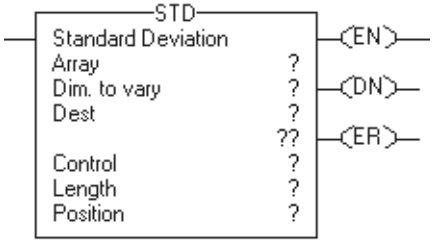
Otros formatos:

Formato:	Sintaxis:
texto neutro	`SRT(array,dim_to_vary,control,length,position);`
texto ASCII	`SRT array dim_to_vary control length position`

Instrucciones relacionadas: AVE, STD

File Standard Deviation (STD)

La instrucción STD es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Arreglo	SINT INT **DINT** **REAL**	tag de arreglo	hallar la desviación estándar de los valores en este arreglo especificar el primer elemento del grupo de elementos que se usan para calcular la desviación estándar **no** use CONTROL.POS en el subíndice
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Dimensión para variar:	DINT	valor inmediato (0, 1, 2)	qué dimensión usar según el número de dimensiones, el orden es: array[dim_0,dim_1,dim_2] array[dim_0,dim_1] array[dim_0]
Destino	REAL	tag	resultado de la operación
Control	CONTROL	tag	estructura de control para la operación
Longitud	DINT	valor inmediato	el número de elementos del arreglo que se usan para calcular el desviación estándar
Posición	DINT	valor inmediato	elemento actual en el arreglo el valor inicial es típicamente 0

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción STD está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando se completa el cálculo.
.ER	BOOL	Se establece el bit de error cuando la instrucción genera un overflow. La instrucción ya no se ejecuta hasta que el programa restablece el bit .ER. La posición del elemento que causó el overflow se almacena en el valor .POS.
.LEN	DINT	La longitud especifica el número de elementos en el arreglo en que la instrucción realiza una operación.
.POS	DINT	La posición contiene la posición del elemento actual al cual la instrucción obtiene acceso.

Descripción:

La instrucción STD calcula la desviación estándar de un conjunto de valores en una dimensión del arreglo y almacena el resultado en el destino.

Importante: Asegúrese de que la longitud no cause que la instrucción exceda la dimensión especificada para variar. Si ocurre esto, el destino será incorrecto. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información.

La desviación estándar se calcula según la fórmula siguiente:

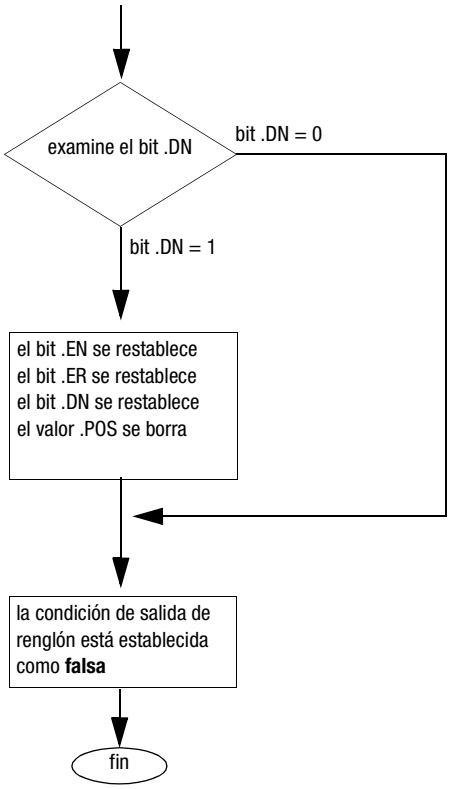
$$\text{Desviación estándar} = \sqrt{\frac{\left(\sum_{i=1}^{N} [\langle X_{(start+i)} - AVE\rangle^2]\right)}{(N-1)}}$$

donde:

- inicio = subíndice de la dimensión para variar del operando del arreglo
- x_i = elemento variable en el arreglo
- N = número de elementos especificados en el arreglo
- AVE =

$$\frac{\left(\sum_{i=1}^{N} x_{(start+i)}\right)}{N}$$

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	
la condición de entrada de renglón es verdadera	La instrucción STD calcula la desviación estándar de los elementos especificados. Internamente, la instrucción usa una instrucción FAL para calcular el promedio. Expresión = cálculo de la desviación estándar Modo = TODOS Para obtener detalles acerca de cómo se ejecuta la instrucción FAL, vea la página 7-8.

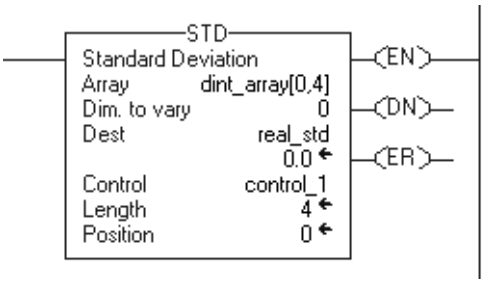
Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
.POS < 0 ó .LEN < 0	4	21
La dimensión que se va a variar no existe para el arreglo especificado	4	20

Ejemplo de STD:

ejemplo 1



dint_array es DINT[4,5]

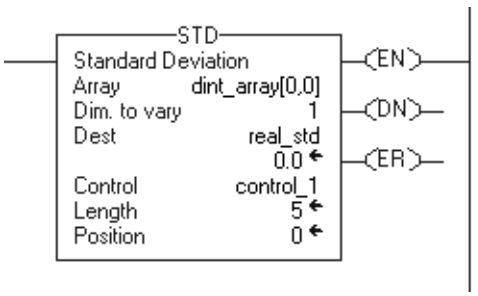
subíndices / dimensión 0	dimensión 1: 0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

$$AVE = \frac{16 + 11 + 6 + 1}{4} = \frac{34}{4} = 8.5$$

$$STD = \sqrt{\frac{\langle 16 - 8.5\rangle^2 + \langle 11 - 8.5\rangle^2 + \langle 6 - 8.5\rangle^2 + \langle 1 - 8.5\rangle^2}{\langle 4 - 1\rangle}} = 6.454972$$

real_std = 6.454972

ejemplo 2



dint_array es DINT[4,5]

subíndices / dimensión 0	dimensión 1: 0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

$$AVE = \frac{20 + 19 + 18 + 17 + 16}{5} = \frac{90}{5} = 18$$

$$STD = \sqrt{\frac{\langle 20 - 18\rangle^2 + \langle 19 - 18\rangle^2 + \langle 18 - 18\rangle^2 + \langle 17 - 18\rangle^2 + \langle 16 - 18\rangle^2}{\langle 5 - 1\rangle}} = 1.581139$$

real_std = 1.581139

Otros formatos:

Formato:	Sintaxis:
texto neutro	`STD(array,dim_to_vary,destination,control,length,position);`
texto ASCII	`STD array dim_to_vary destination control length position`

Instrucciones relacionadas: AVE, SRT

Notas:

Instrucciones de arreglo (archivo)/ desplazamiento
(BSL, BSR, FFL, FFU, LFL, LFU)

Introducción

Use las instrucciones de arreglo (archivo)/desplazamiento para modificar la ubicación de datos dentro de los arreglos.

Si usted desea:	Use esta instrucción:	Vea la página:
Cargar bits en, desplazar bits a través de y descarga bits desde un arreglo de bit, bit tras bit.	BSL	8-2
	BSR	8-5
Cargar y descargar valores en el mismo orden.	FFL	8-8
	FFU	8-14
Cargar y descargar valores en el orden invertido.	LFL	8-20
	LFU	8-26

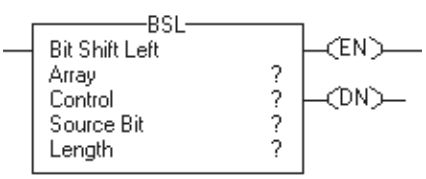
Usted puede combinar los tipos de datos, pero esto puede resultar en la pérdida de precisión y errores de redondeo.

Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Bit Shift Left (BSL)

La instrucción BSL es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Arreglo	DINT	tag de arreglo	arreglo que se modifica especificar el primer elemento del grupo de elementos **no** use CONTROL.POS en el subíndice
Control	CONTROL	tag	estructura de control para la operación
Bit de origen	BOOL	tag	bit que se desplaza
Longitud	DINT	valor inmediato	número de bits en el arreglo que se desplazan

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción BSL está habilitada.
.DN	BOOL	Se establece el bit de efectuado para indicar que los bits se han desplazado una posición a la izquierda.
.UL	BOOL	El bit de descarga es la salida de la instrucción. El bit .UL almacena el estado del bit que se desplazó fuera del rango de los bits.
.ER	BOOL	Se establece el bit de error cuando .LEN < 0.
.LEN	DINT	La longitud especifica el número de bits de arreglo que se desplazan.

Descripción:

La instrucción BSL desplaza los bits especificados dentro del arreglo una posición a la izquierda. Una vez habilitada, la instrucción descarga el bit superior de los bits especificados al bit .UL, desplaza el resto de los bits una posición a la izquierda y carga el bit de origen en el bit 0 del arreglo.

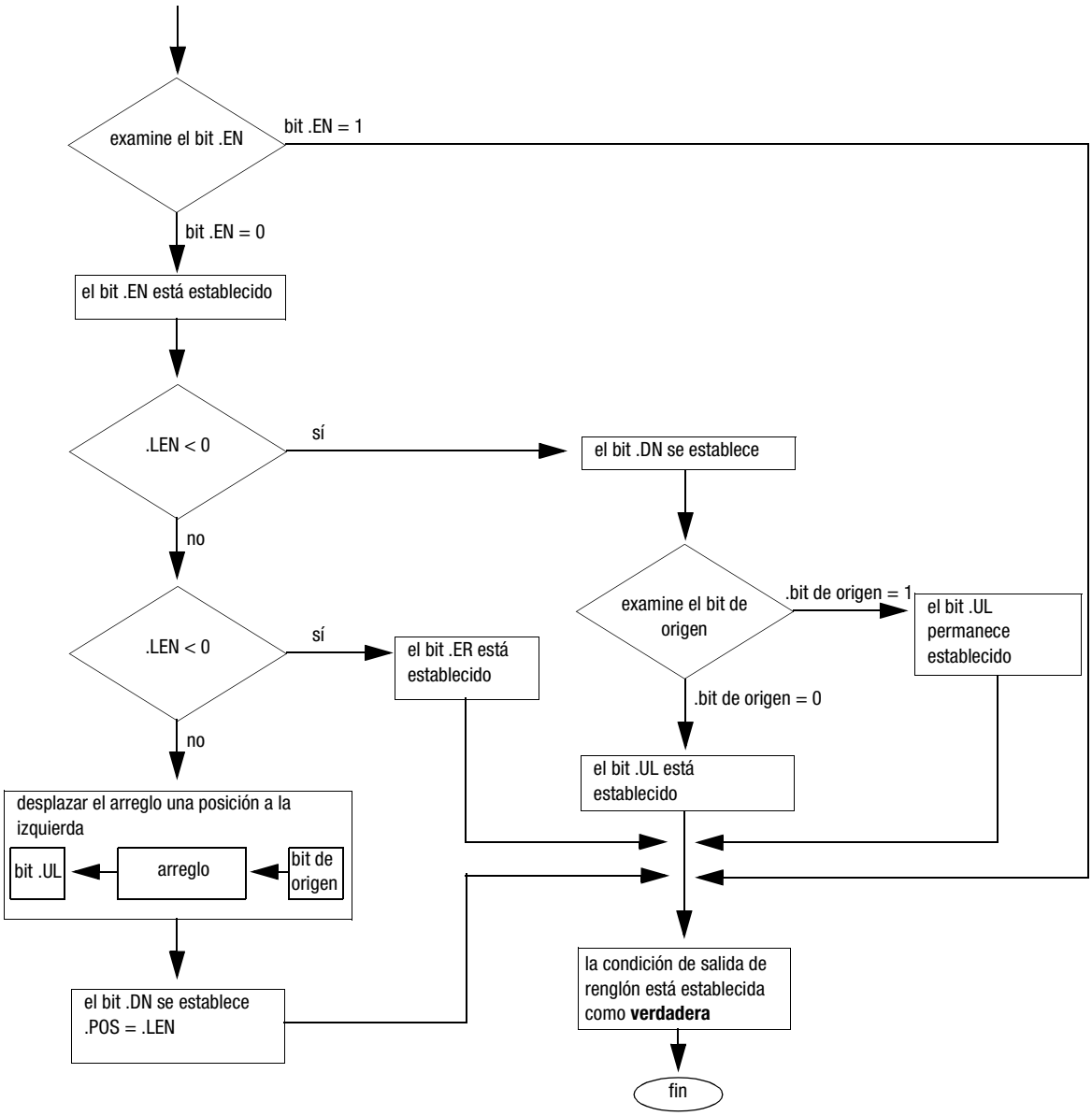
La instrucción BSL realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. El valor .POS se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. El valor .POS se restablece. La condición de salida de renglón se establece como falsa.

Condición: **Acción:**

la condición de entrada de renglón es verdadera

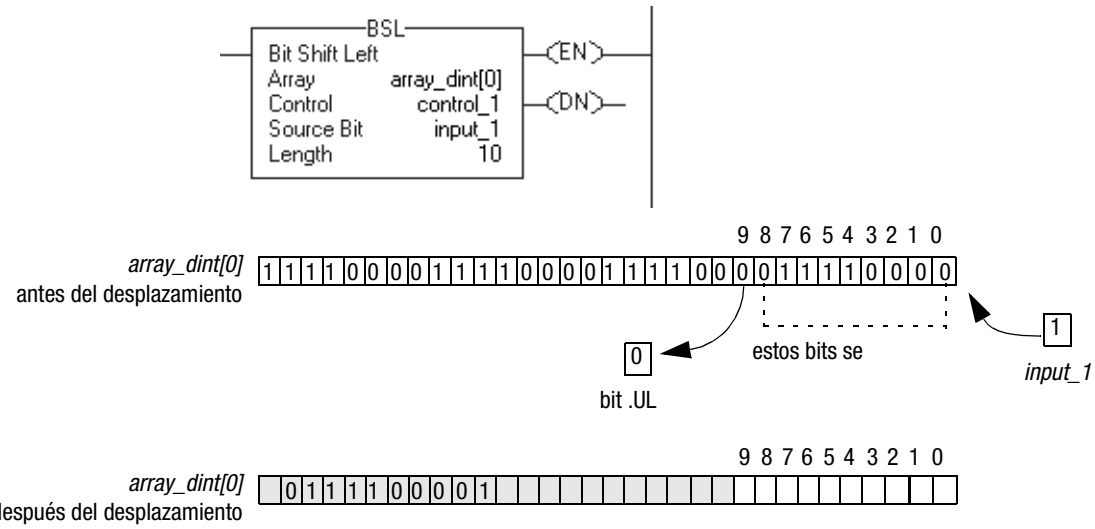

examine el bit .EN
bit .EN = 1
bit .EN = 0
el bit .EN está establecido
.LEN < 0
sí
el bit .DN se establece
no
examine el bit de origen
.bit de origen = 1
el bit .UL permanece establecido
.LEN < 0
sí
el bit .ER está establecido
.bit de origen = 0
no
el bit .UL está establecido
desplazar el arreglo una posición a la izquierda
bit .UL
arreglo
bit de origen
la condición de salida de renglón está establecida como **verdadera**
el bit .DN se establece .POS = .LEN
fin

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

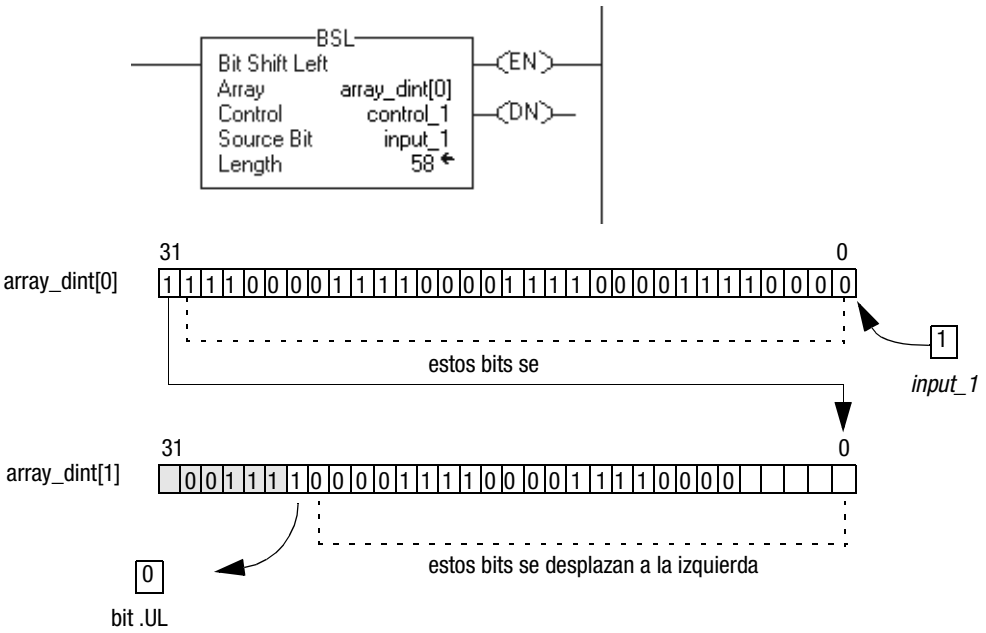
Ejemplo de BSL:

ejemplo 1


BSL
Bit Shift Left
Array array_dint[0]
Control control_1
Source Bit input_1
Length 10
EN
DN
9 8 7 6 5 4 3 2 1 0
array_dint[0]
antes del desplazamiento
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
0
estos bits se
1
input_1
bit .UL
9 8 7 6 5 4 3 2 1 0
array_dint[0]
después del desplazamiento
0 1 1 1 1 0 0 0 0 1

Una vez habilitada, la instrucción BSL comienza a partir del bit 0 en *array_dint[0].* La instrucción descarga *array_dint[0].9* en el bit .UL, desplaza el resto de los bits y carga *input_1* en *array_dint[0].0*. Los valores en el resto de los bits (10-31) no son válidos.

ejemplo 2


BSL
Bit Shift Left
Array array_dint[0]
Control control_1
Source Bit input_1
Length 58
EN
DN
31
0
array_dint[0]
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
estos bits se
1
input_1
31
0
array_dint[1]
0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
estos bits se desplazan a la izquierda
0
bit .UL

Una vez habilitada, la instrucción BSL comienza a partir del bit 0 en *array_dint[0].* La instrucción descarga *array_dint[1].25* en el bit .UL, desplaza el resto de los bits y carga *input_1* en *array_dint[0].0*. Los valores en el resto de los bits (31-26 en *array_dint[1]*) no son válidos. Observe cómo *array_dint[0].31* se desplaza a través de las palabras a *array_dint[1].0*.

Otros formatos:

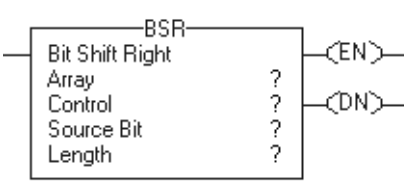
Formato:	Sintaxis:
texto neutro	`BSL(array,control,source_bit,length);`
texto ASCII	`BSL array control source_bit length`

Instrucciones relacionadas: BSR

Bit Shift Right (BSR)

La instrucción BSR es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Arreglo	DINT	tag de arreglo	arreglo que se modifica especificar el elemento a partir del cual se inicia el desplazamiento **no** use CONTROL.POS en el subíndice
Control	CONTROL	tag	estructura de control para la operación
Bit de origen	BOOL	tag	bit que se desplaza
Longitud	DINT	valor inmediato	número de bits en el arreglo que se desplazan

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción BSR está habilitada.
.DN	BOOL	Se establece el bit de efectuado para indicar que los bits se han desplazado una posición a la derecha.
.UL	BOOL	El bit de descarga es la salida de la instrucción. El bit .UL almacena el estado del bit que se desplazó fuera del rango de los bits.
.ER	BOOL	Se establece el bit de error cuando .LEN < 0.
.LEN	DINT	La longitud especifica el número de bits de arreglo que se desplazan.

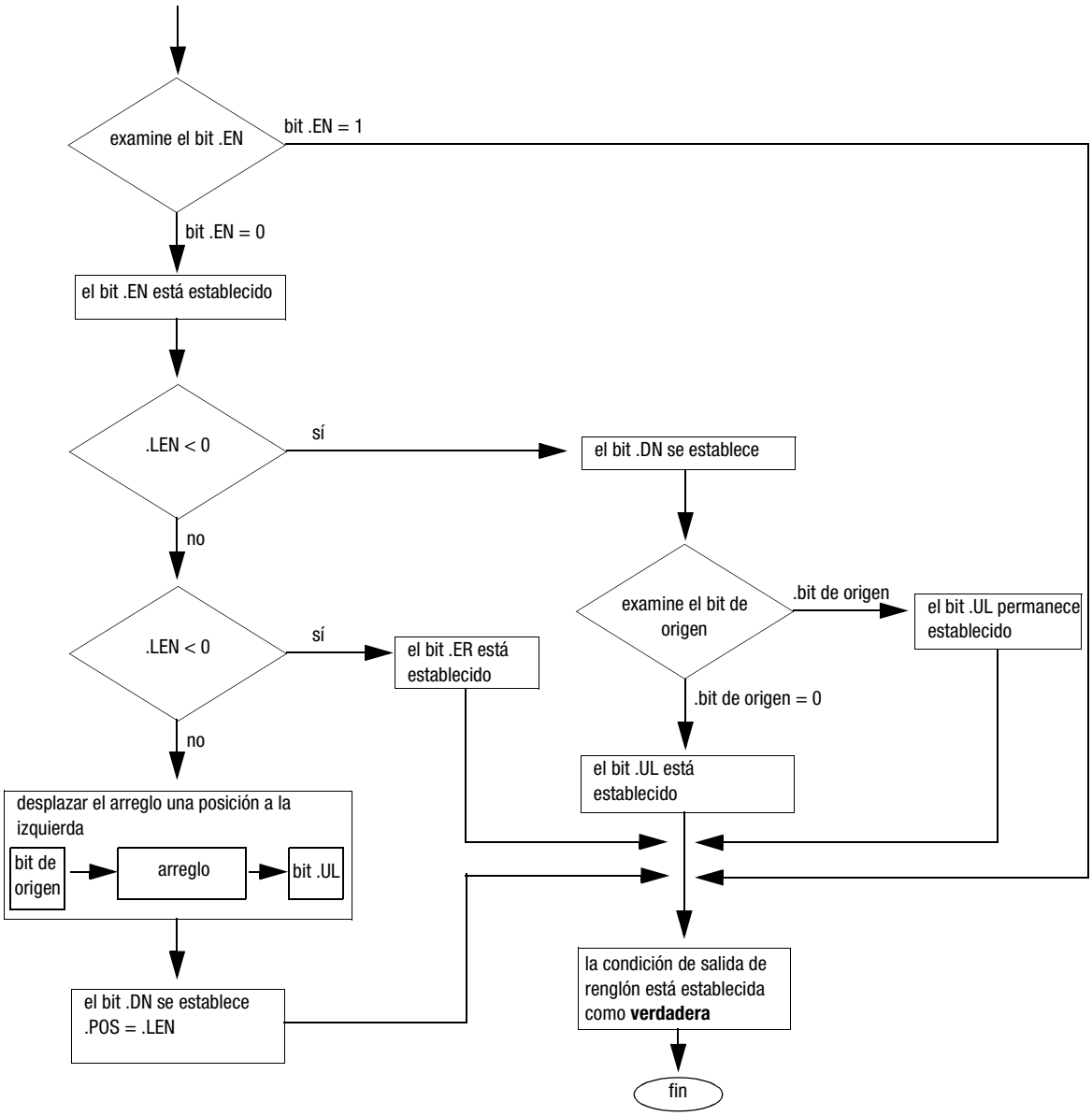
Descripción: La instrucción BSR desplaza los bits especificados dentro del arreglo una posición a la derecha. Una vez habilitada, la instrucción descarga el valor en el bit 0 del arreglo al bit .UL, desplaza el resto de los bits una posición a la derecha y carga el bit de origen en el bit superior de los bits especificados.

La instrucción BSR realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

Ejecución:

Condición:	Acción:
preescán	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. El valor .POS se restablece. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit .EN se restablece. El bit .DN se restablece. El bit .ER se restablece. El valor .POS se restablece. La condición de salida de renglón se establece como falsa.

Condición:	**Acción:**
la condición de entrada de renglón es verdadera	

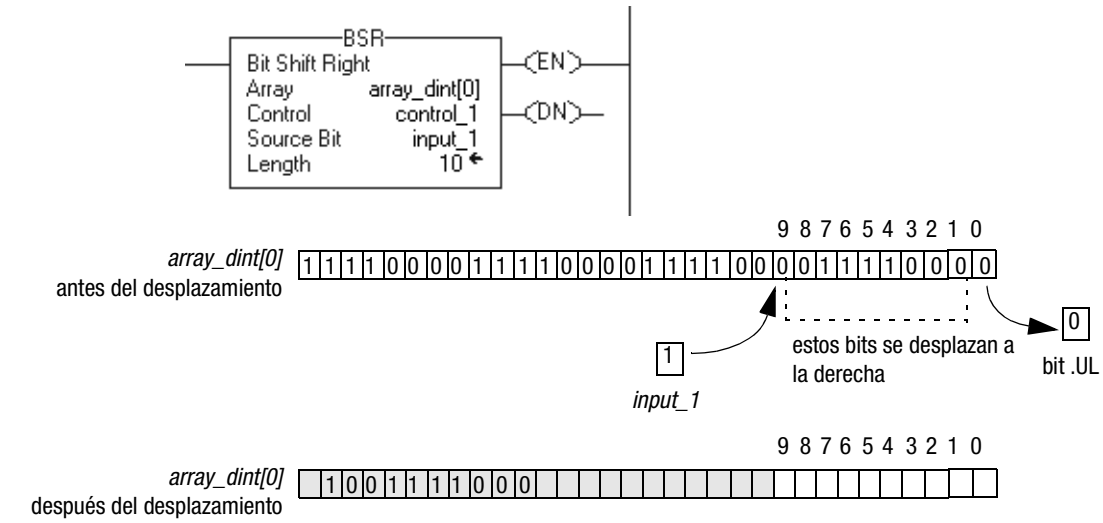


Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

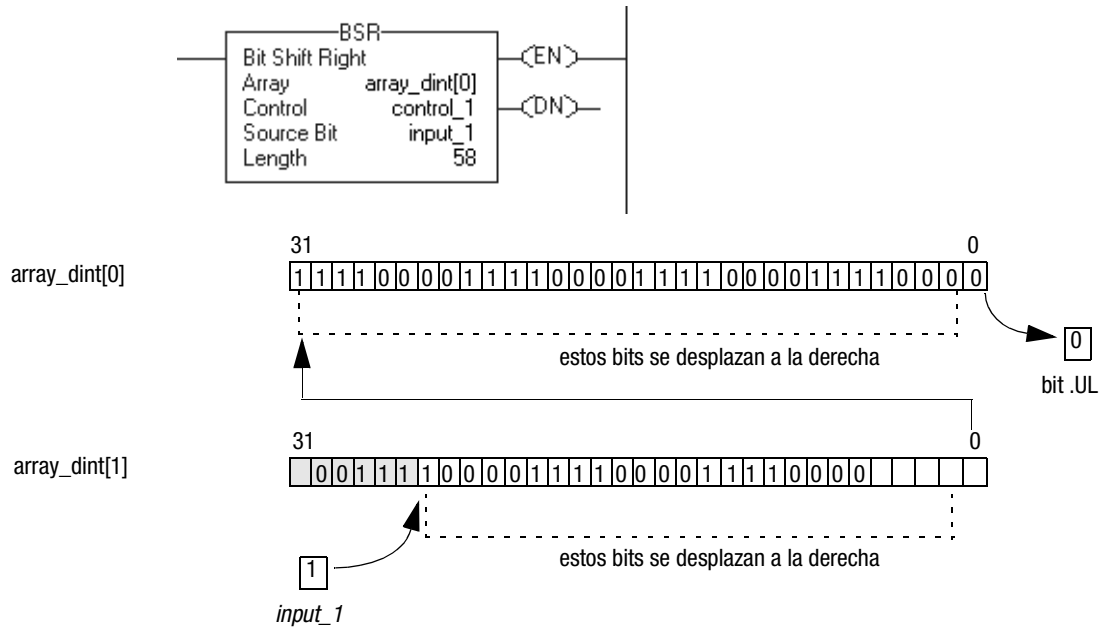
Ejemplo de BSR:

ejemplo 1


BSR
Bit Shift Right
Array array_dint[0]
Control control_1
Source Bit input_1
Length 10
EN
DN
9 8 7 6 5 4 3 2 1 0
array_dint[0]
antes del desplazamiento
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
0
estos bits se desplazan a la derecha
bit .UL
1
input_1
9 8 7 6 5 4 3 2 1 0
array_dint[0]
después del desplazamiento
1 0 0 1 1 1 1 0 0 0

Una vez habilitada, la instrucción BSR comienza a partir del bit 9 en *array_dint[0].* La instrucción descarga *array_dint[0].0* en el bit .UL, desplaza el resto de los bits a la derecha y carga *input_1* en *array_dint[0].9.* Los valores en el resto de los bits (10 – 31) no son válidos.

ejemplo 2


BSR
Bit Shift Right
Array array_dint[0]
Control control_1
Source Bit input_1
Length 58
EN
DN
31
0
array_dint[0]
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
0
estos bits se desplazan a la derecha
bit .UL
31
0
array_dint[1]
0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
estos bits se desplazan a la derecha
1
input_1

Una vez habilitada, la instrucción BSR comienza a partir del bit 25 en *array_dint[1].* La instrucción descarga *array_dint[0].0* en el bit .UL, desplaza el resto de los bits a la derecha y carga *input_1* en *array_dint[1].25.* Los valores en el resto de los bits (31 – 26 en *dint_array[1]*) no son válidos Observe cómo *array_dint[1].0* se desplaza a través de las palabras a *array_dint[0].31*.

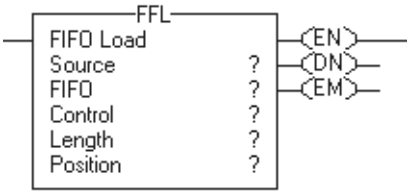
Otros formatos:

Formato:	Sintaxis:
texto neutro	BSR(*array,control,source_bit,length*);
texto ASCII	BSR *array control source_bit length*

Instrucciones relacionadas: BSL

FIFO Load (FFL)

La instrucción FFL es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	**SINT** **INT** **DINT** **REAL** **estructura**	valor inmediato tag	datos que se almacenan en FIFO
	El origen se convierte al tipo de datos del tag de arreglo. Un número entero menor se convierte en un número entero mayor según la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
FIFO	**SINT** **INT** **DINT** **REAL** **estructura**	tag de arreglo	FIFO que se modifica especificar el primer elemento FIFO **no** use CONTROL.POS en el subíndice
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL que el FFU asociado
Longitud	DINT	valor inmediato	número máximo de elementos que FIFO puede contener a la vez
Posición	DINT	valor inmediato	La próxima ubicación en FIFO donde la instrucción carga datos el valor inicial es típicamente 0

Si usted usa una estructura definida por el usuario como el tipo de datos para el operando de origen o FIFO, use la misma estructura para los dos operandos.

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción FFL está habilitada.
.DN	BOOL	Se establece el bit de efectuado para indicar que FIFO está lleno (.POS = .LEN). El bit .DN inhibe la carga de FIFO hasta que .POS < .LEN.
.EM	BOOL	El bit de vacío indica que FIFO está vacío. Si .LEN ≤ 0 ó .POS < 0, se establecen los bits .EM y .DN.
.LEN	DINT	La longitud especifica el número máximo de elementos que FIFO puede contener a la vez.
.POS	DINT	La posición identifica la ubicación en FIFO donde la instrucción cargará el próximo valor.

Descripción: La instrucción FFL copia el valor de origen a FIFO. Use la instrucción FFL con una instrucción FFU para almacenar y recuperar datos según un orden de primero en entrar/primero en salir. Cuando se usan en parejas, las instrucciones FFL y FFU establecen un registro de desplazamiento asíncrono.

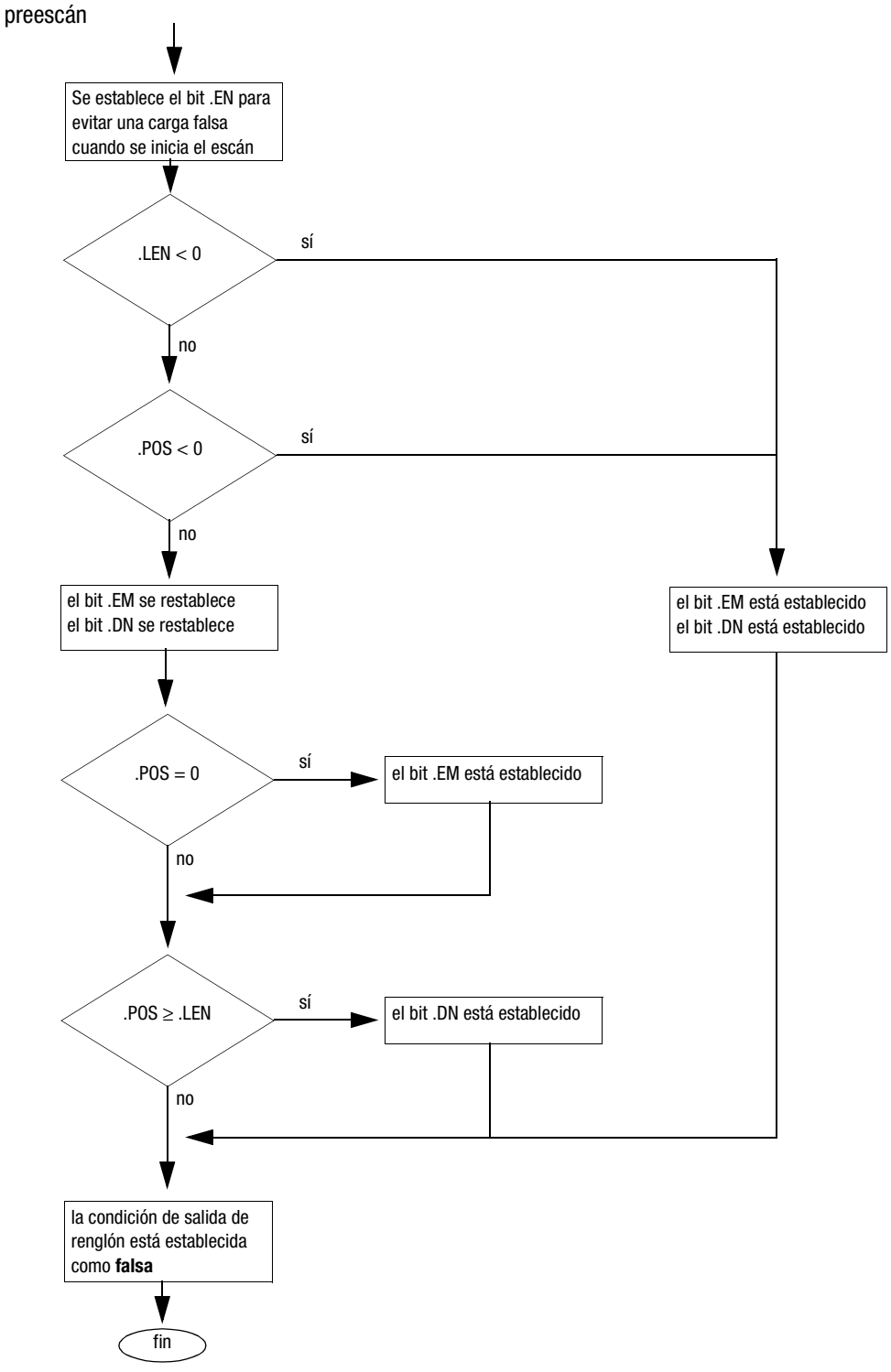
Típicamente el origen y el FIFO son del mismo tipo de datos.

Una vez habilitada, la instrucción FFL carga el valor de origen a la posición en el FIFO identificado por el valor .POS. La instrucción carga un valor cada vez que la instrucción se habilita hasta que el FIFO esté lleno.

La instrucción FFL realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

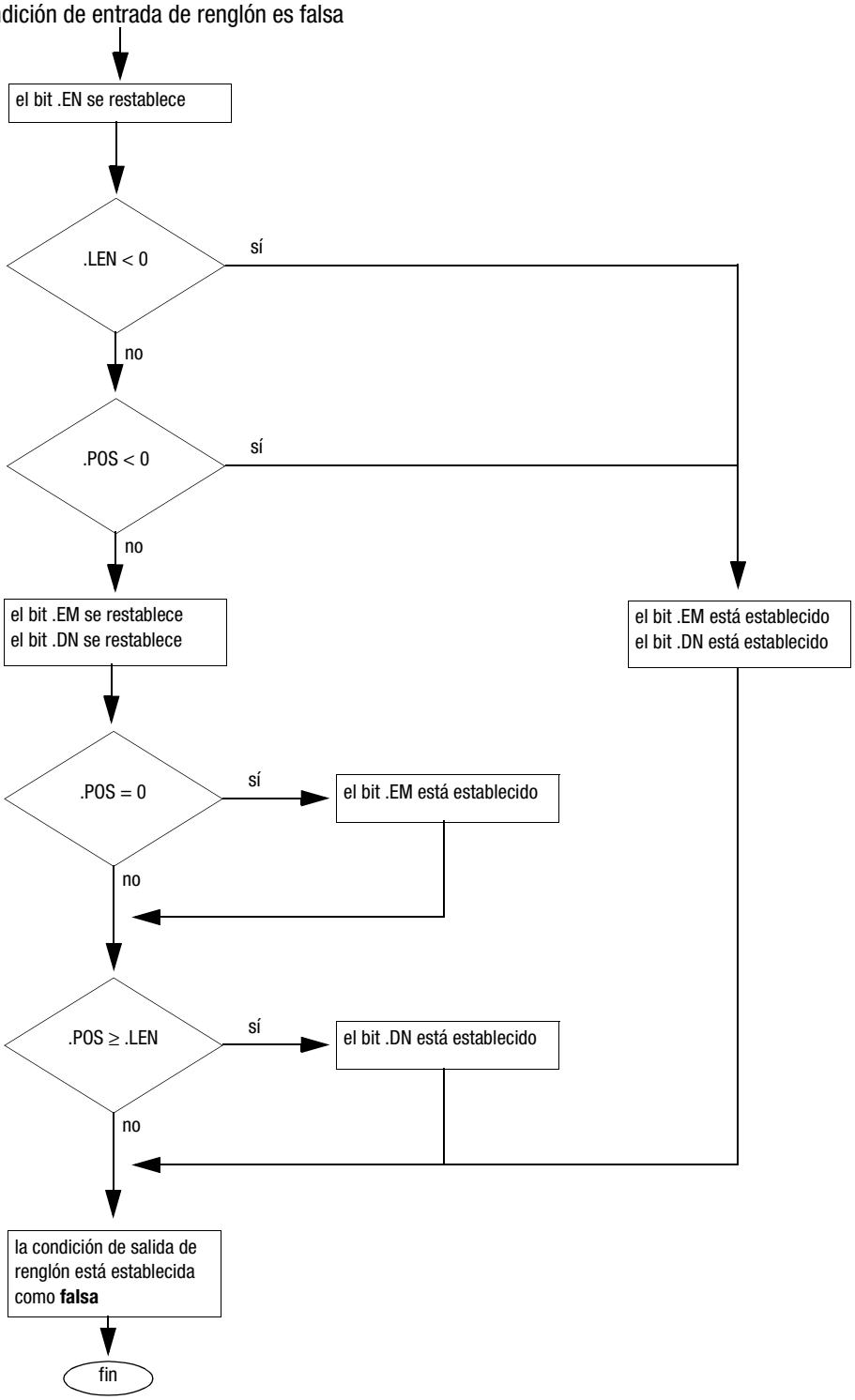
Ejecución:

Condición: **Acción:**



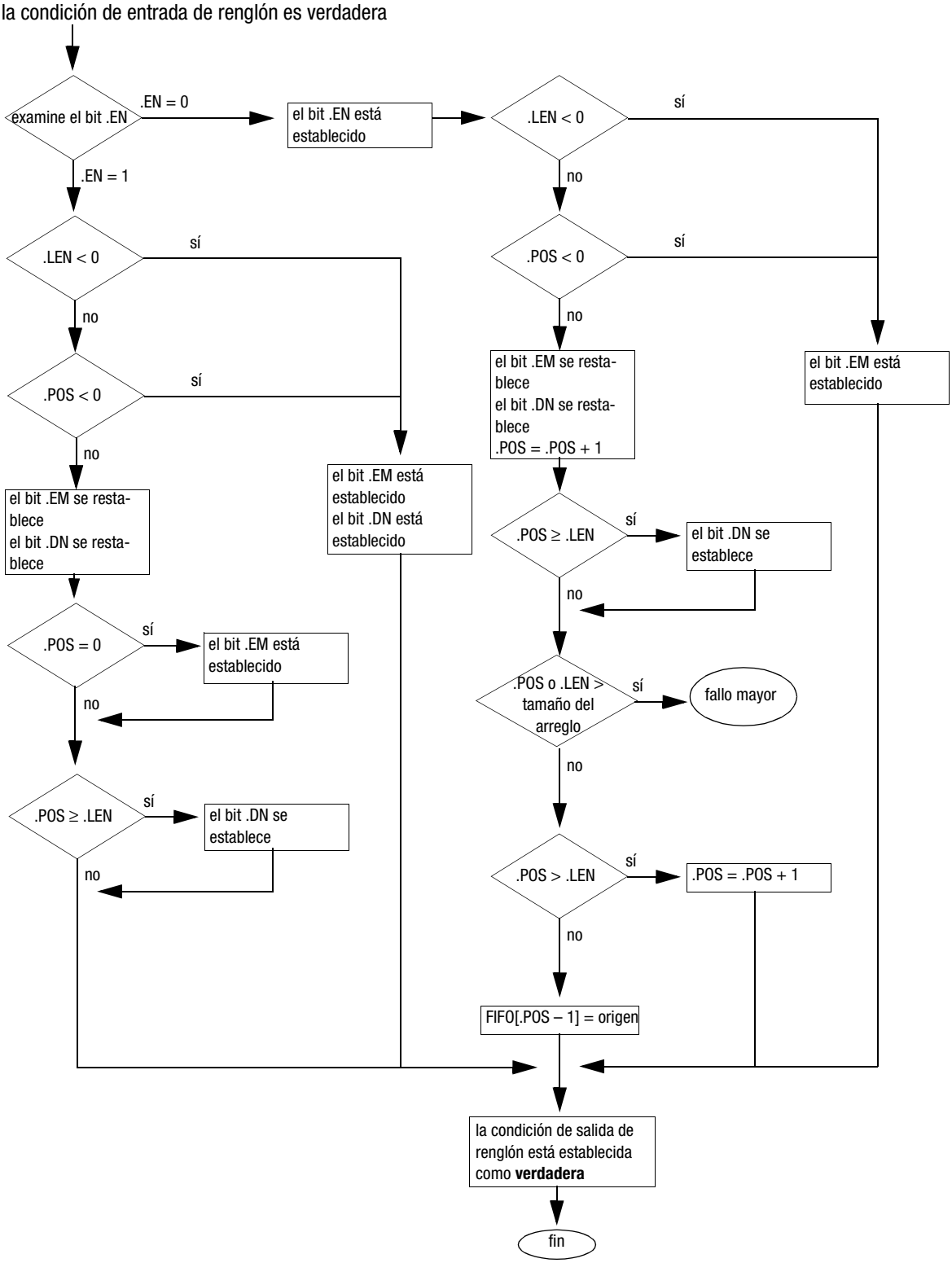
Condición: **Acción:**

la condición de entrada de renglón es falsa



Condición: **Acción:**

la condición de entrada de renglón es verdadera

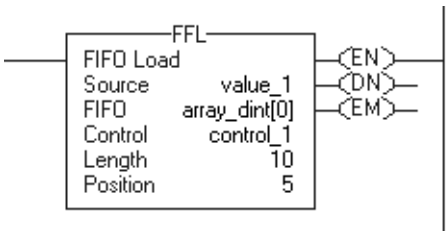


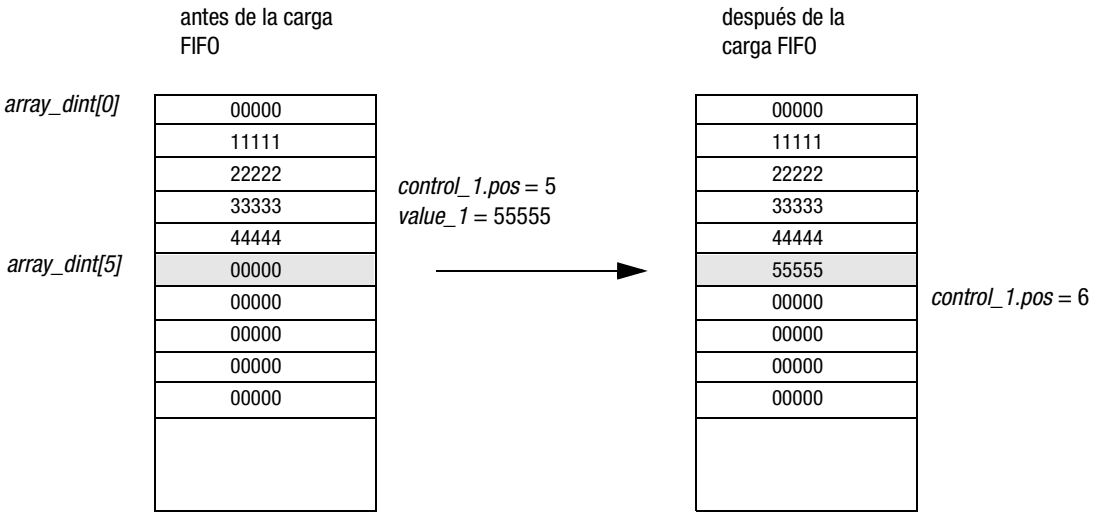
Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
(elemento inicial + .POS) > tamaño del arreglo FIFO	4	20

Ejemplo de FFL:





Una vez habilitada, la instrucción FFL carga *value_1* en la próxima posición en el FIFO, la cual es *array_dint[5]* en este ejemplo.

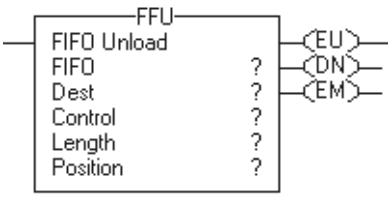
Otros formatos:

Formato:	Sintaxis:
texto neutro	`FFL(source,FIFO,control,length,position);`
texto ASCII	`FFL source FIFO control length position`

Instrucciones relacionadas: FFU, LFL, LFU

FIFO Unload (FFU)

La instrucción FFU es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
FIFO	**SINT** **INT** **DINT** **REAL** **estructura**	tag de arreglo	FIFO que se modifica especificar el primer elemento FIFO **no** use CONTROL.POS en el subíndice
Destino	**SINT** **INT** **DINT** **REAL** **estructura**	tag	valor que sale del FIFO
	El valor de destino se convierte al tipo de datos del tag de destino. Un número entero menor se convierte en un número entero mayor según la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL que el FFL asociado
Longitud	DINT	valor inmediato	número máximo de elementos que FIFO puede contener a la vez
Posición	DINT	valor inmediato	La próxima ubicación en FIFO donde la instrucción descarga datos el valor inicial es típicamente 0

Si usted usa una estructura definida por el usuario como el tipo de datos para el operando FIFO o de destino, use la misma estructura para los dos operandos.

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EU	BOOL	El bit de descarga habilitación indica que la instrucción FFU está habilitada. El bit .EU se establece para impedir una descarga falsa cuando se inicia el escán del programa.
.DN	BOOL	Se establece el bit de efectuado para indicar que FIFO está lleno (.POS = .LEN).
.EM	BOOL	El bit de vacío indica que FIFO está vacío. Si .LEN ≤ 0 ó .POS < 0, se establecen los bits .EM y .DN.
.LEN	DINT	La longitud especifica el número máximo de elementos en FIFO.
.POS	DINT	La posición identifica el fin de los datos que se han cargado en FIFO.

Descripción: La instrucción FFU descarga el valor desde la posición 0 (primera posición) de FIFO y almacena dicho valor en el destino. El resto de los datos en l FIFO se desplaza hacia abajo una posición. Use la instrucción FFU con una instrucción FFL para almacenar y recuperar datos según un orden de primero en entrar/primero en salir.

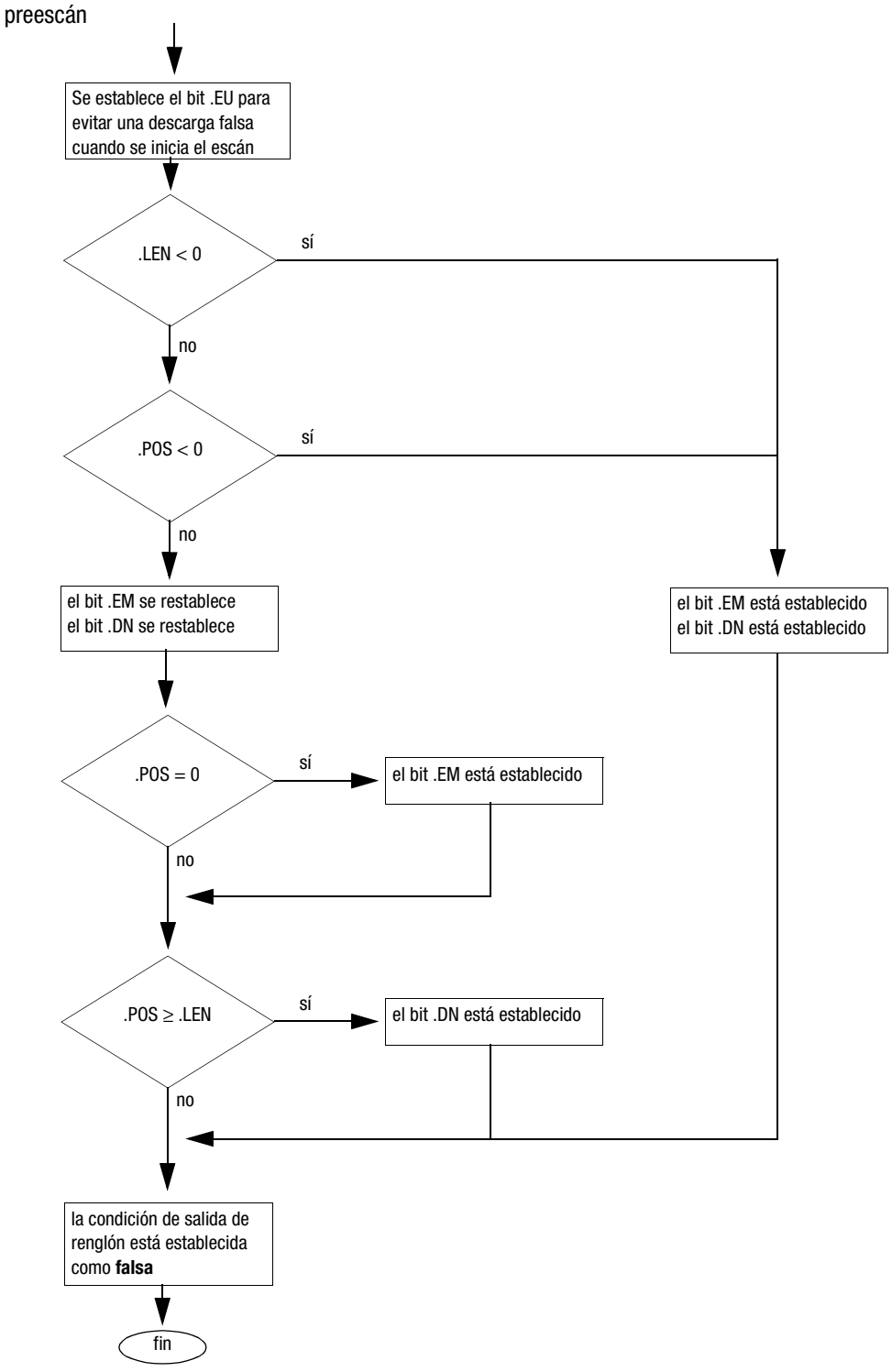
Si FIFO es del tipo de datos DINT, el destino debe ser un tipo de datos DINT; si FIFO es del tipo de datos REAL, el destino debe ser un tipo de datos REAL.

Una vez habilitada, la instrucción FFU descarga los datos desde el primer elemento de FIFO y coloca dicho valor en el destino. La instrucción descarga un valor cada vez que la instrucción se habilita hasta que FIFO esté vacío. Si FIFO está vacío, FFU retorna la cifra 0 al destino.

La instrucción FFU realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información.

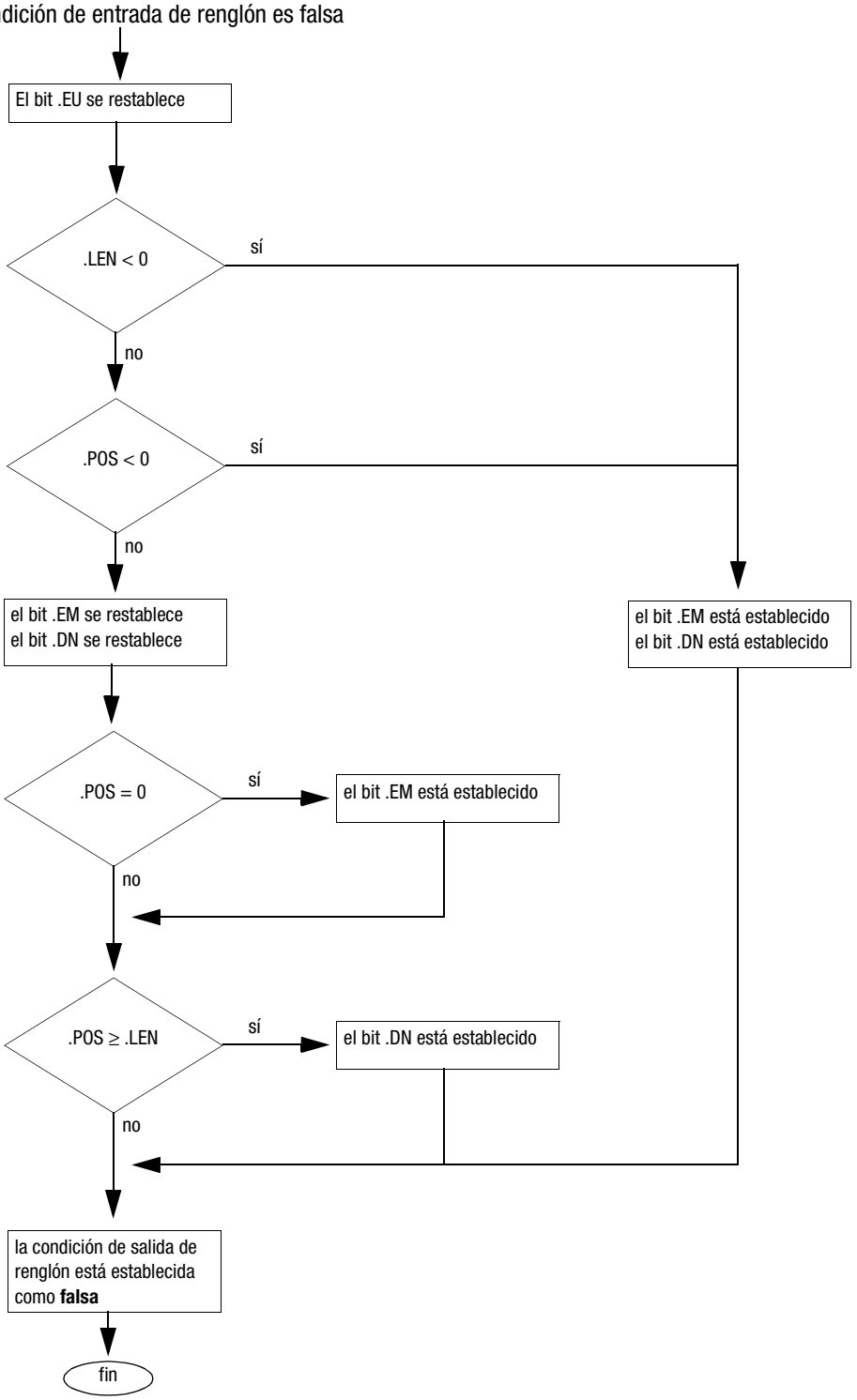
Ejecución:

Condición: **Acción:**



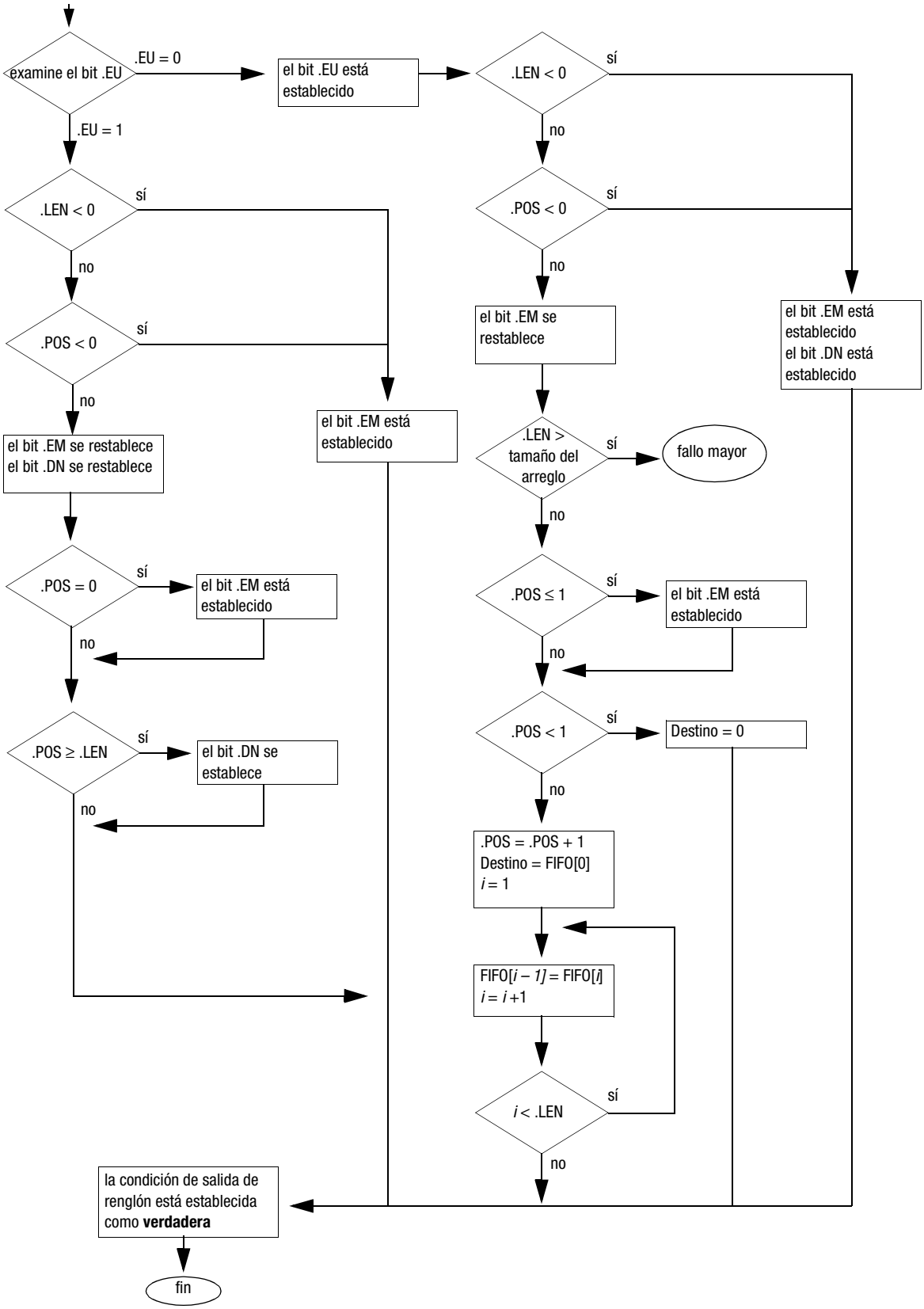
Condición:	**Acción:**

la condición de entrada de renglón es falsa



Condición: **Acción:**

la condición de entrada de renglón es verdadera

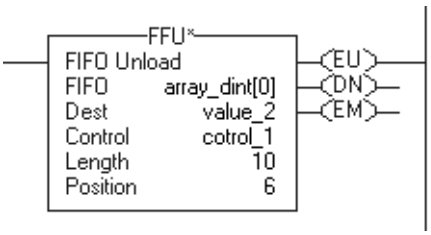


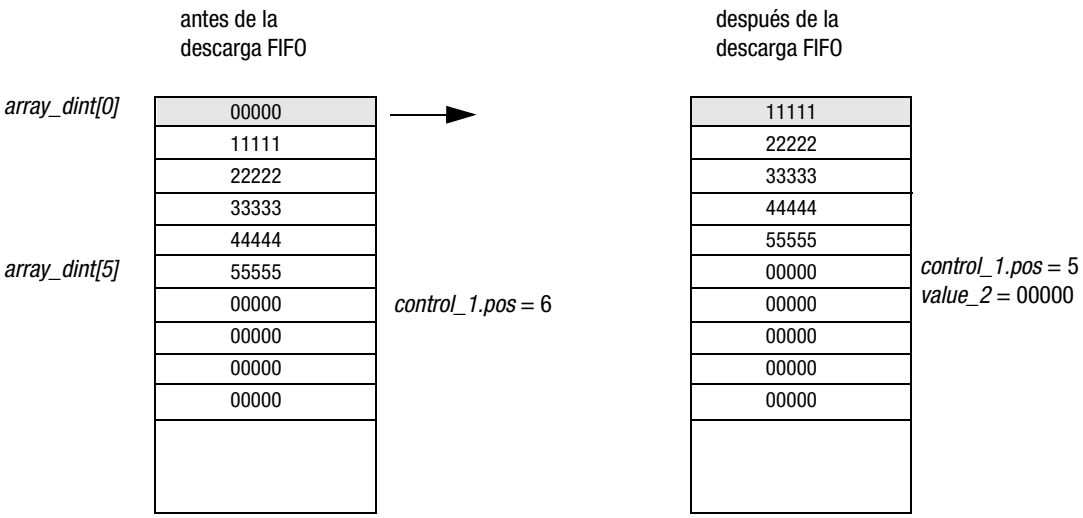
Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
Longitud > tamaño del arreglo FIFO	4	20

Ejemplo de FFU:





Una vez habilitada, la instrucción FFU descarga *array_dint[0]* en *value_2* y desplaza el resto de los elementos en *array_dint*.

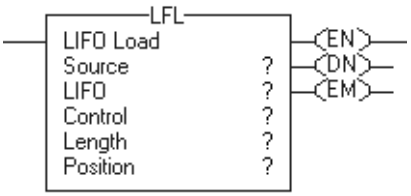
Otros formatos:

Formato:	Sintaxis:
texto neutro	`FFU(FIFO,destination,control,length,position);`
texto ASCII	`FFU FIFO destination control length position`

Instrucciones relacionadas: FFL, LFL, LFU

LIFO Load (LFL)

La instrucción LFL es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Origen	**SINT** **INT** **DINT** **REAL** **estructura**	valor inmediato tag	datos que se almacenan en LIFO
	El origen se convierte al tipo de datos del tag de arreglo. Un número entero menor se convierte en un número entero mayor según la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
LIFO	**SINT** **INT** **DINT** **REAL** **estructura**	tag de arreglo	LIFO que se modifica especificar el primer elemento de LIFO **no** use CONTROL.POS en el subíndice
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL que el LFU asociado
Longitud	DINT	valor inmediato	número máximo de elementos que LIFO puede contener a la vez
Posición	DINT	valor inmediato	La próxima ubicación en LIFO donde la instrucción carga datos el valor inicial es típicamente 0

Si usted usa una estructura definida por el usuario como el tipo de datos para el operando de origen o LIFO, use la misma estructura para los dos operandos.

Estructura de control:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción LFL está habilitada.
.DN	BOOL	Se establece el bit de efectuado para indicar que LIFO está lleno (.POS = .LEN). El bit .DN inhibe la carga de LIFO hasta que .POS < .LEN.
.EM	BOOL	El bit de vacío indica que LIFO está vacío. Si .LEN ≤ 0 ó .POS < 0, se establecen los bits .EM y .DN.
.LEN	DINT	La longitud especifica el número máximo de elementos que LIFO puede contener a la vez.
.POS	DINT	La posición identifica la ubicación en LIFO donde la instrucción cargará el próximo valor.

Descripción: La instrucción LFL copia el valor de origen a LIFO. Use la instrucción LFL con una instrucción LFU para almacenar y recuperar datos según un orden de último en entrar/primero en salir. Cuando se usan en parejas, las instrucciones LFL y LFU establecen un registro de desplazamiento asíncrono.

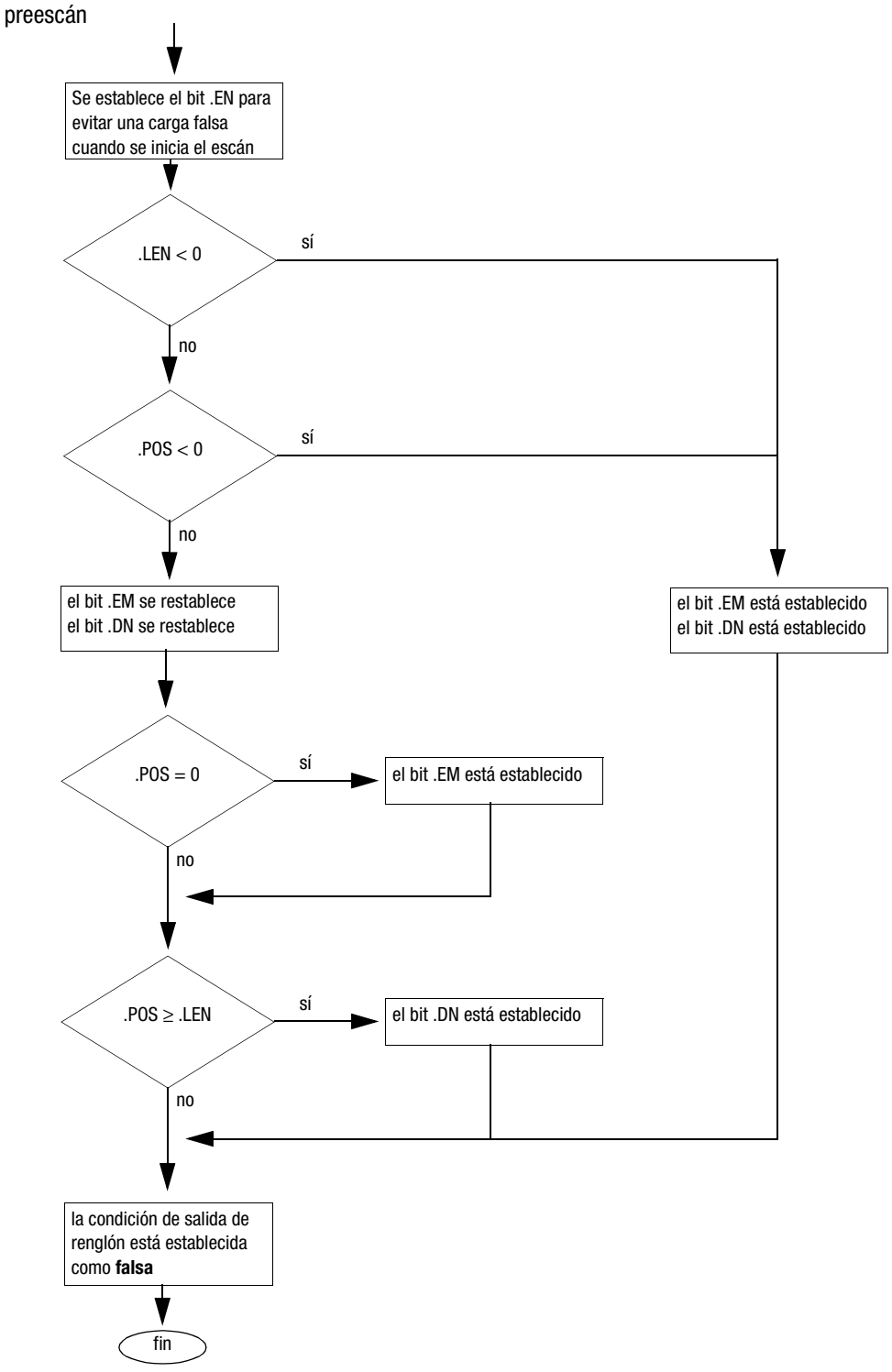
Típicamente el origen y LIFO son del mismo tipo de datos.

Una vez habilitada, la instrucción LFL carga el valor de origen a la posición en LIFO identificada por el valor .POS. La instrucción carga un valor cada vez que la instrucción se habilita hasta que LIFO esté lleno.

La instrucción LFL realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

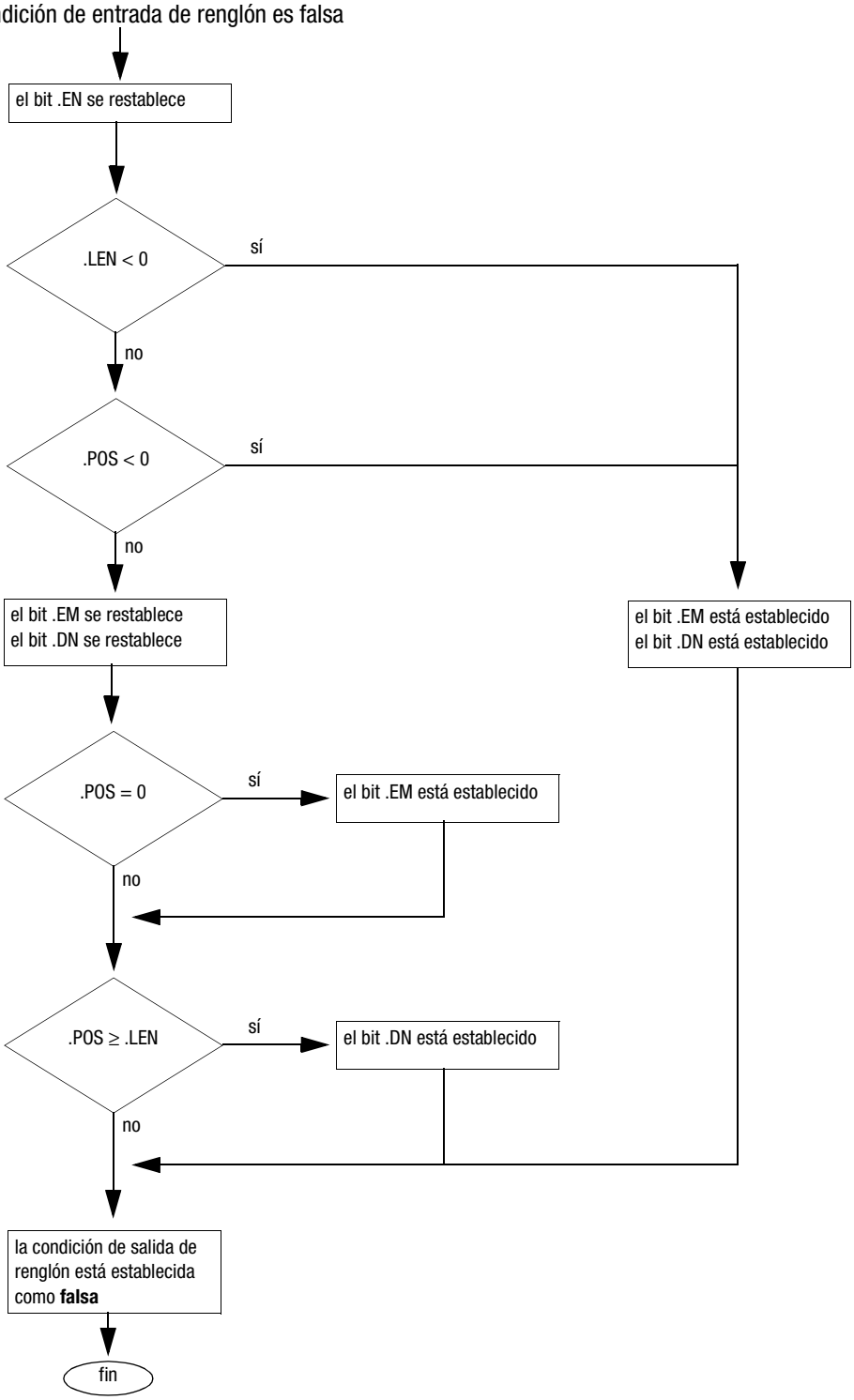
Ejecución:

Condición: **Acción:**

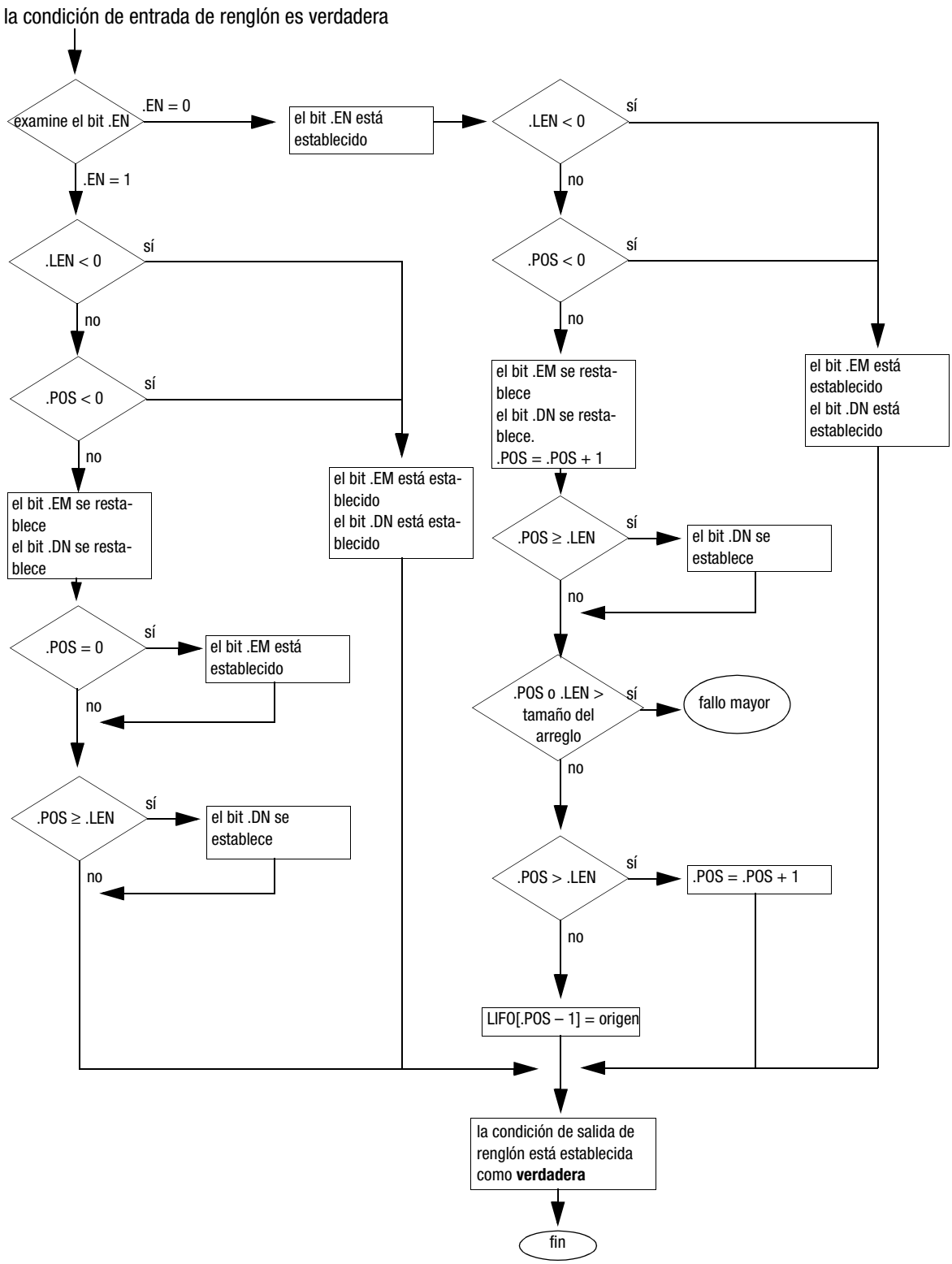


Condición: **Acción:**

la condición de entrada de renglón es falsa



Condición: **Acción:**

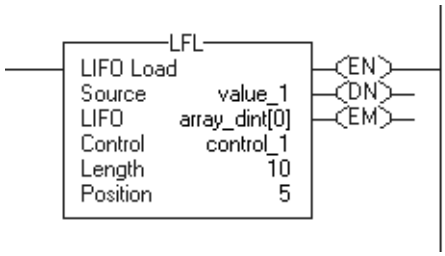


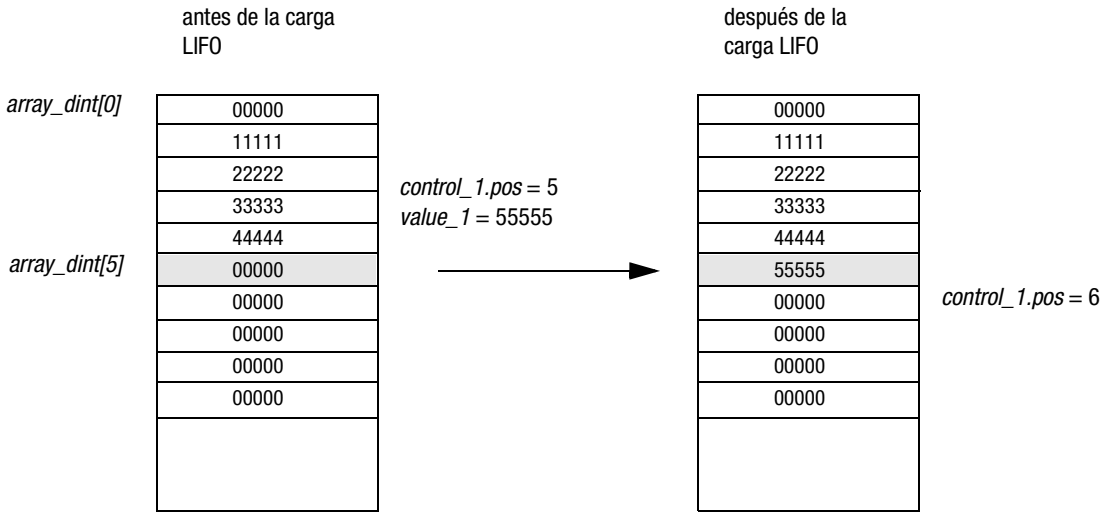
Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
(elemento inicial + .POS) > tamaño del arreglo LIFO	4	20

Ejemplo de LFL:





Una vez habilitada, la instrucción LFL carga *value_1* a la próxima posición en LIFO, la cual es *array_dint[5]* en este ejemplo.

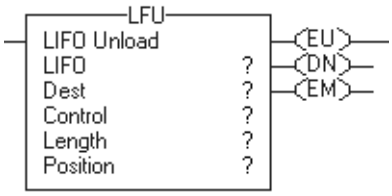
Otros formatos:

Formato:	Sintaxis:
texto neutro	`LFL(source,LIFO,control,length,position);`
texto ASCII	`LFL source LIFO control length position`

Instrucciones relacionadas: LFU, FFL, FFU

LIFO Unload (LFU)

La instrucción LFU es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
LIFO	**SINT** **INT** **DINT** **REAL** **estructura**	tag de arreglo	LIFO que se modifica especificar el primer elemento LIFO **no** use CONTROL.POS en el subíndice
Destino	**SINT** **INT** **DINT** **REAL** **estructura**	tag	valor que sale del LIFO El valor de destino se convierte al tipo de datos del tag de destino. Un número entero menor se convierte en un número entero mayor según la extensión con signo. Consulte “Conversiones de datos” en la página A-6.
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL que el LFL asociado
Longitud	DINT	valor inmediato	número máximo de elementos que LIFO puede contener a la vez
Posición	DINT	valor inmediato	La próxima ubicación en LIFO donde la instrucción descarga datos el valor inicial es típicamente 0

Si usted usa una estructura definida por el usuario como el tipo de datos para el operando LIFO o de destino, use la misma estructura para los dos operandos.

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EU	BOOL	El bit de habilitación de descarga indica que la instrucción LFU está habilitada. El bit .EU se establece para impedir una descarga falsa cuando se inicia el escán del programa.
.DN	BOOL	Se establece el bit de efectuado para indicar que LIFO está lleno (.POS = .LEN).
.EM	BOOL	El bit de vacío indica que LIFO está vacío. Si .LEN ≤ 0 ó .POS < 0, se establecen los bits .EM y .DN.
.LEN	DINT	La longitud especifica el número máximo de elementos que LIFO puede contener a la vez.
.POS	DINT	La posición identifica el fin de los datos que se han cargado en el LIFO.

Descripción: La instrucción LFU descarga el valor en .POS de LIFO y almacena 0 en ese lugar. Use la instrucción LFU con una instrucción LFL para almacenar y recuperar datos según un orden de último en entrar/primero en salir.

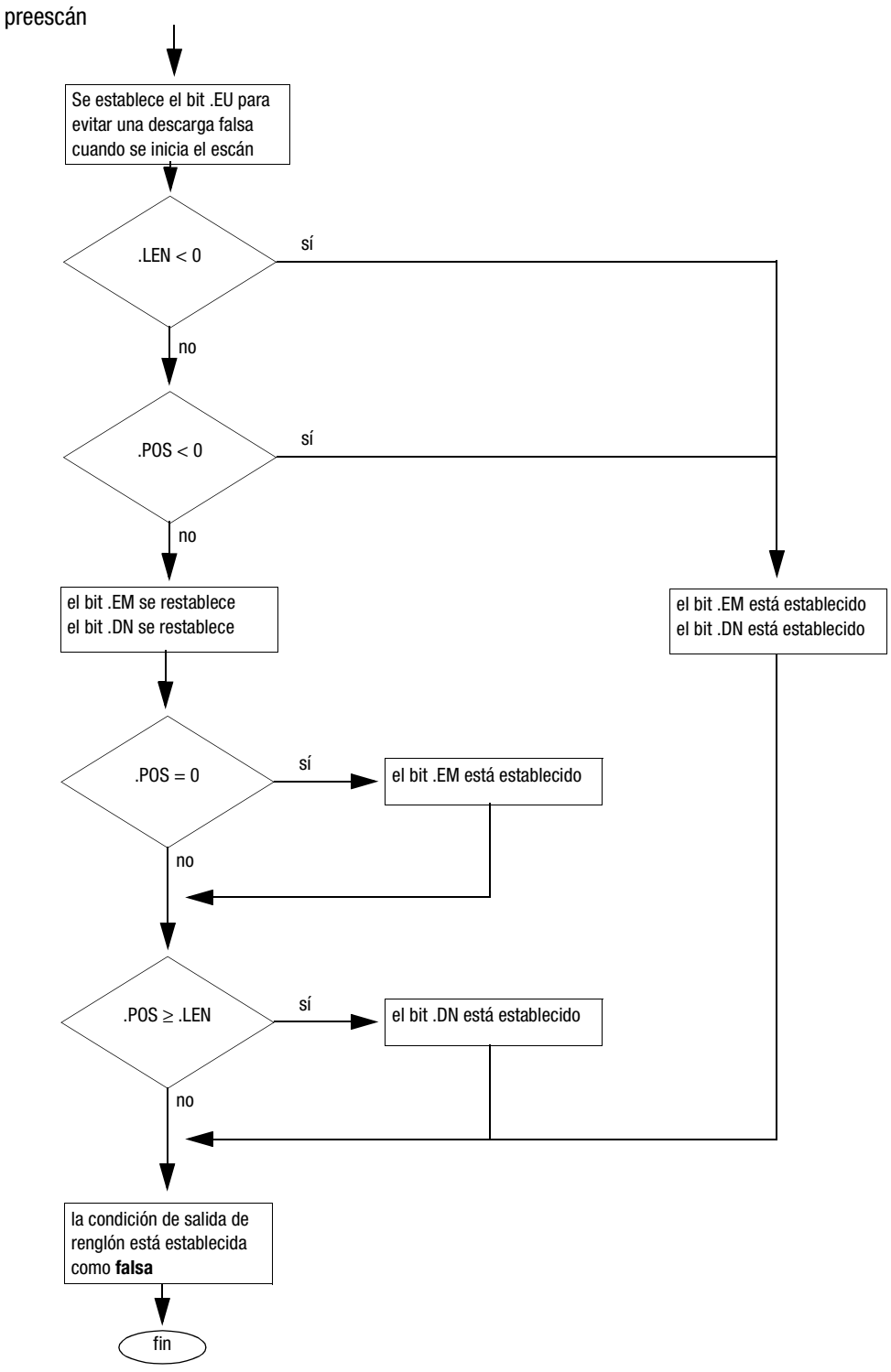
Si LIFO es del tipo de datos DINT, el destino debe ser un tipo de datos DINT; si LIFO es del tipo de datos REAL, el destino debe ser un tipo de datos REAL.

Una vez habilitada, la instrucción LFU descarga el valor de .POS de LIFO y coloca dicho valor en el destino. La instrucción descarga un valor y lo reemplaza con 0 cada vez que se habilita la instrucción hasta que LIFO esté vacío. Si LIFO está vacío, la instrucción LFU retorna la cifra 0 al destino.

La instrucción LFU realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

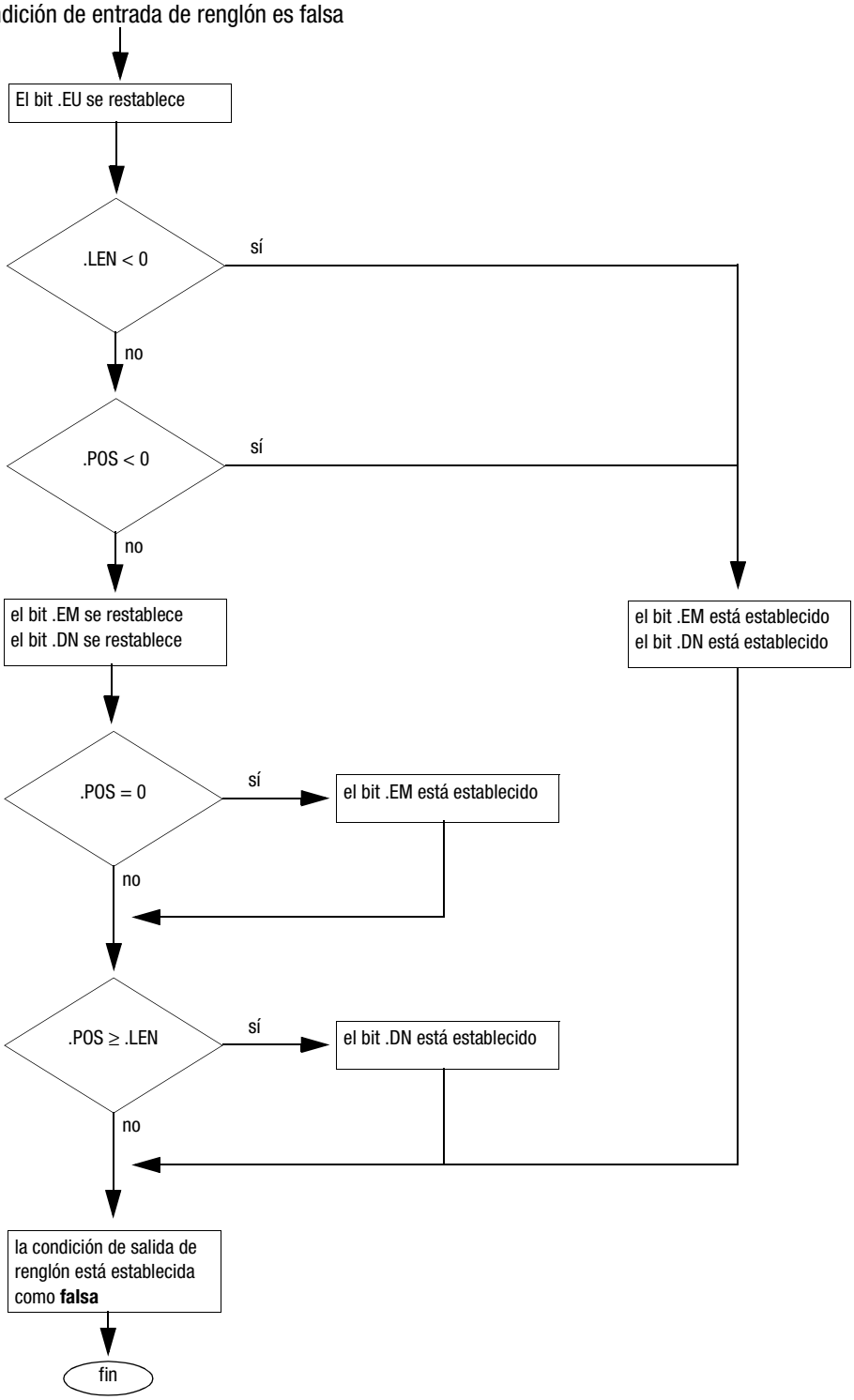
Ejecución:

Condición: **Acción:**



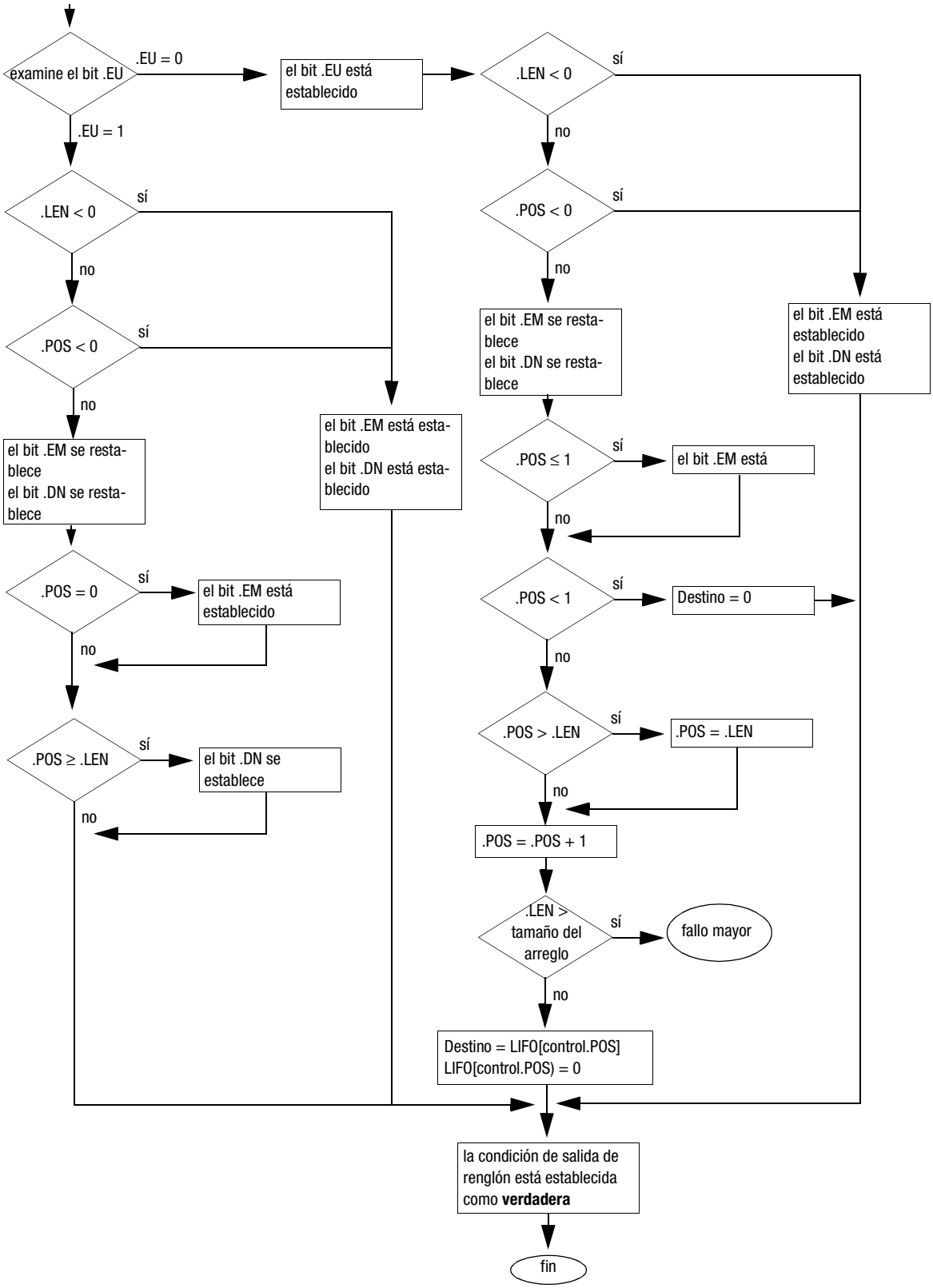
Condición: **Acción:**

la condición de entrada de renglón es falsa



Condición: **Acción:**

la condición de entrada de renglón es verdadera

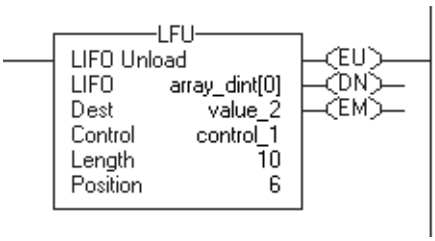


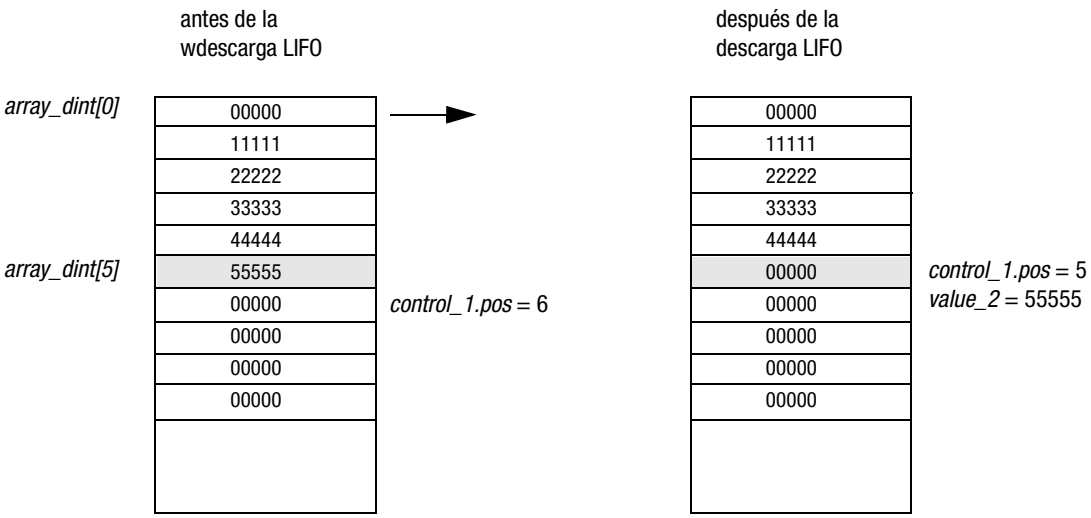
Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
Longitud > tamaño del arreglo LIFO	4	20

Ejemplo de LFU:





Una vez habilitada, la instrucción LFU descarga *array_dint[5]* en *value_2*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`LFU(LIFO,destination,control,length,position);`
texto ASCII	`LFU LIFO destination control length position`

Instrucciones relacionadas: LFU, FFL, FFU

Notas:

Instrucciones de secuenciador
(SQI, SQO, SQL)

Introducción

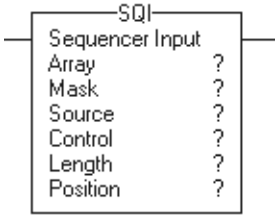
Las instrucciones de secuenciador monitorean las operaciones uniformes y repetibles.

Si usted desea:	Use esta instrucción:	Vea la página:
Detectar cuándo un paso se ha completado.	SQI	9-2
Establecer las condiciones de salida para el próximo paso.	SQO	9-6
Cargar condiciones de referencia en los arreglos del secuenciador.	SQL	9-11

Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Sequencer Input (SQI)

La instrucción SQI es una instrucción de entrada.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Arreglo	DINT	tag de arreglo	arreglo de secuenciador especifique el primer elemento del arreglo de secuenciador **no** use CONTROL.POS en el subíndice
Máscara	SINT INT **DINT**	tag valor inmediato	qué bits se bloquean o se pasan
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Origen	SINT INT DINT	tag	datos de entrada para el arreglo de secuenciador
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL usado con instrucciones SQO y SQL
Longitud	DINT	valor inmediato	número de elementos en el arreglo (tabla de secuenciadores) que se comparan
Posición	DINT	valor inmediato	posición actual en el arreglo el valor inicial es típicamente 0

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.ER	BOOL	Se establece el bit de error cuando .LEN $\leq$ 0, .POS < 0 ó .POS > .LEN.
.LEN	DINT	La longitud especifica el número de pasos en el arreglo de secuenciador.
.POS	DINT	La posición identifica el elemento que la instrucción compara actualmente.

Descripción:

La instrucción SQI detecta cuándo se completa un paso en un pareja secuencial de instrucciones SQO/SQI. Una vez habilitada, la instrucción SQI compara un elemento de origen mediante una máscara con un elemento del arreglo para ver si hay igualdad.

Generalmente se usa la misma estructura CONTROL usada con las instrucciones SQO y SQL.

La instrucción SQI realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

Cómo introducir un valor de máscara inmediato

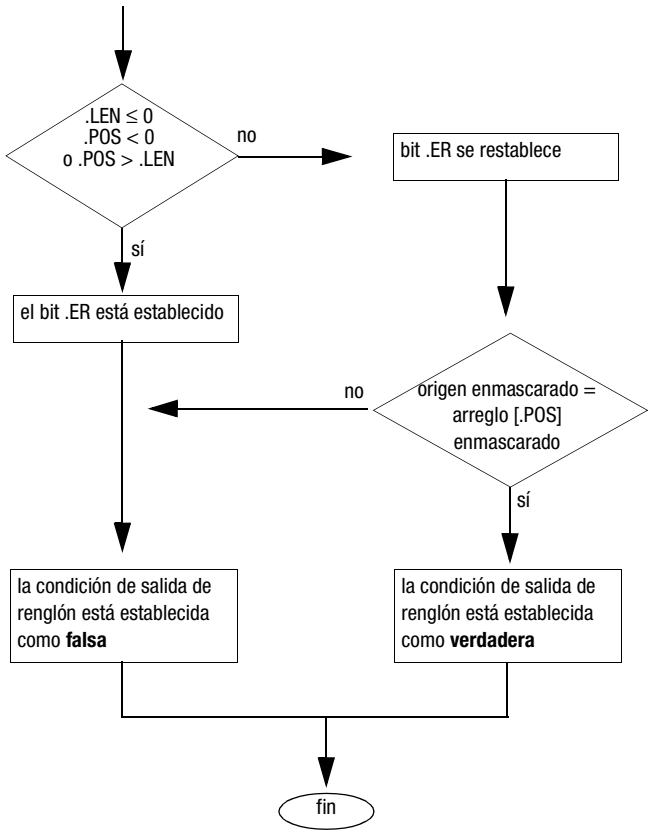
Cuando se introduce una máscara, el software de programación tiene valores decimales predeterminados. Si desea introducir una máscara usando otro formato, ponga el prefijo correcto antes del valor.

Prefijo:	Descripción:
16#	hexadecimal por ejemplo; 16#0F0F
8#	octal por ejemplo; 8#16
2#	binario por ejemplo; 2#00110011

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.

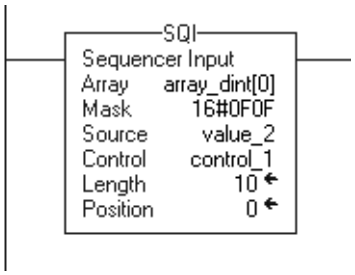
la condición de entrada de renglón es verdadera



Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de SQI:



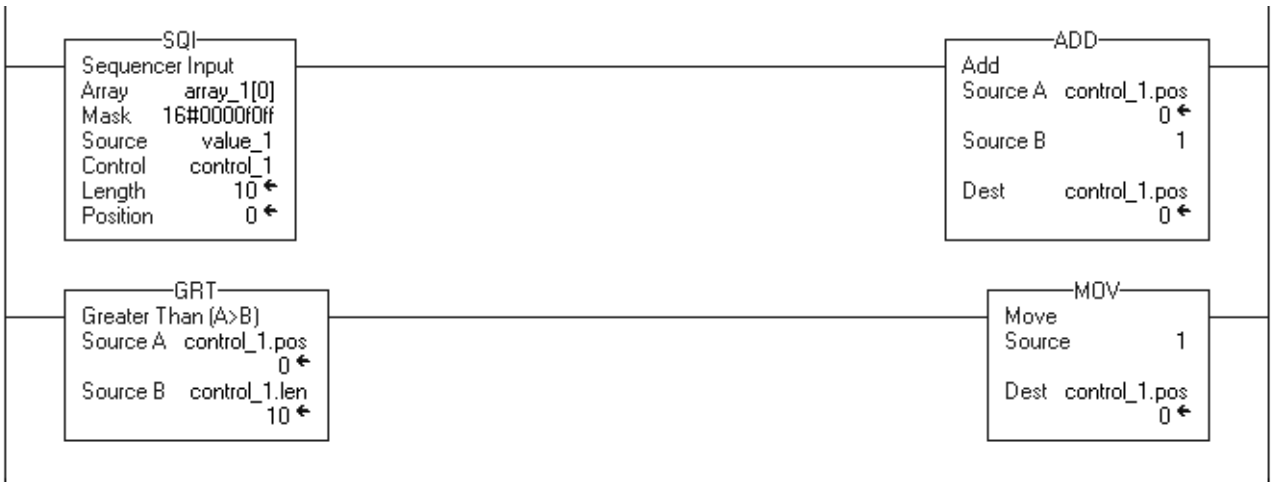
Una vez habilitada, la instrucción SQI pasa *value_2* a través de la máscara para determinar si el resultado es igual al elemento actual en *array_dint*. La comparación con máscara es verdadera, por lo tanto la condición de salida de renglón se hace verdadera.

Operando SQI:	Ejemplos de valores (DINT mostrados en la pantalla en formato binario):
Origen	xxxxxxxx xxxxxxxx xxxx0101 xxxx1010
Máscara	00000000 00000000 00001111 00001111
Arreglo	xxxxxxxx xxxxxxxx xxxx0101 xxxx1010

Una cifra 0 en la máscara significa que el bit no se compara (designado por xxxx en este ejemplo).

Cómo usar SQI sin SQO

Si usted usa la instrucción SQI sin una instrucción SQO como pareja, es necesario incrementar externamente el arreglo de secuenciador.



La instrucción SQI compara el valor de origen.
La instrucción ADD incrementa el arreglo de secuenciador.
La instrucción GRT determina si hay otro valor disponible para verificar el arreglo de secuenciador.
La instrucción MOV restablece el valor de posición después de pasar por el arreglo de secuenciador una vez.

Otros formatos:

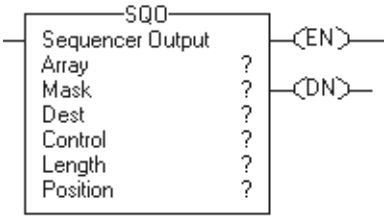
Formato:	Sintaxis:
texto neutro	`SQI(array,mask,source,control,length,position);`
texto ASCII	`SQI array mask source control length position`

Instrucciones relacionadas: SQO, SQL

Sequencer Output (SQO)

La instrucción SQO es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Arreglo	DINT	tag de arreglo	arreglo de secuenciador especifique el primer elemento del arreglo de secuenciador **no** use CONTROL.POS en el subíndice
Máscara	SINT INT **DINT**	tag valor inmediato	qué bits se bloquean o se pasan
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Destino	DINT	tag	datos de salida desde el arreglo de secuenciador
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL usado con las instrucciones SQI y SQL
Longitud	DINT	valor inmediato	número de elementos en el arreglo (tabla de secuenciadores) de los cuales se establecen salidas
Posición	DINT	valor inmediato	posición actual en el arreglo el valor inicial es típicamente 0

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción SQO está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando todos los elementos especificados han sido movidos al destino.
.ER	BOOL	Se establece el bit de error cuando .LEN $\leq$ 0, .POS < 0 ó .POS > .LEN.
.LEN	DINT	La longitud especifica el número de pasos en el arreglo de secuenciador.
.POS	DINT	La posición identifica el elemento que el controlador manipula actualmente.

Descripción: La instrucción SQO establece las condiciones de salida para el próximo paso de una pareja secuencial de las instrucciones SQO/SQI. Una vez habilitada, la instrucción SQO incrementa la posición, mueve los datos en la posición mediante la máscara y almacena el resultado en el destino. Si .POS > .LEN, la instrucción da la vuelta al inicio del arreglo de secuenciador y continúa con .POS = 1.

Generalmente se usa la misma estructura CONTROL usada con las instrucciones SQI y SQL.

La instrucción SQO realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

Cómo introducir un valor de máscara inmediato

Cuando se introduce una máscara, el software de programación tiene valores decimales predeterminados. Si desea introducir una máscara usando otro formato, ponga el prefijo correcto antes del valor.

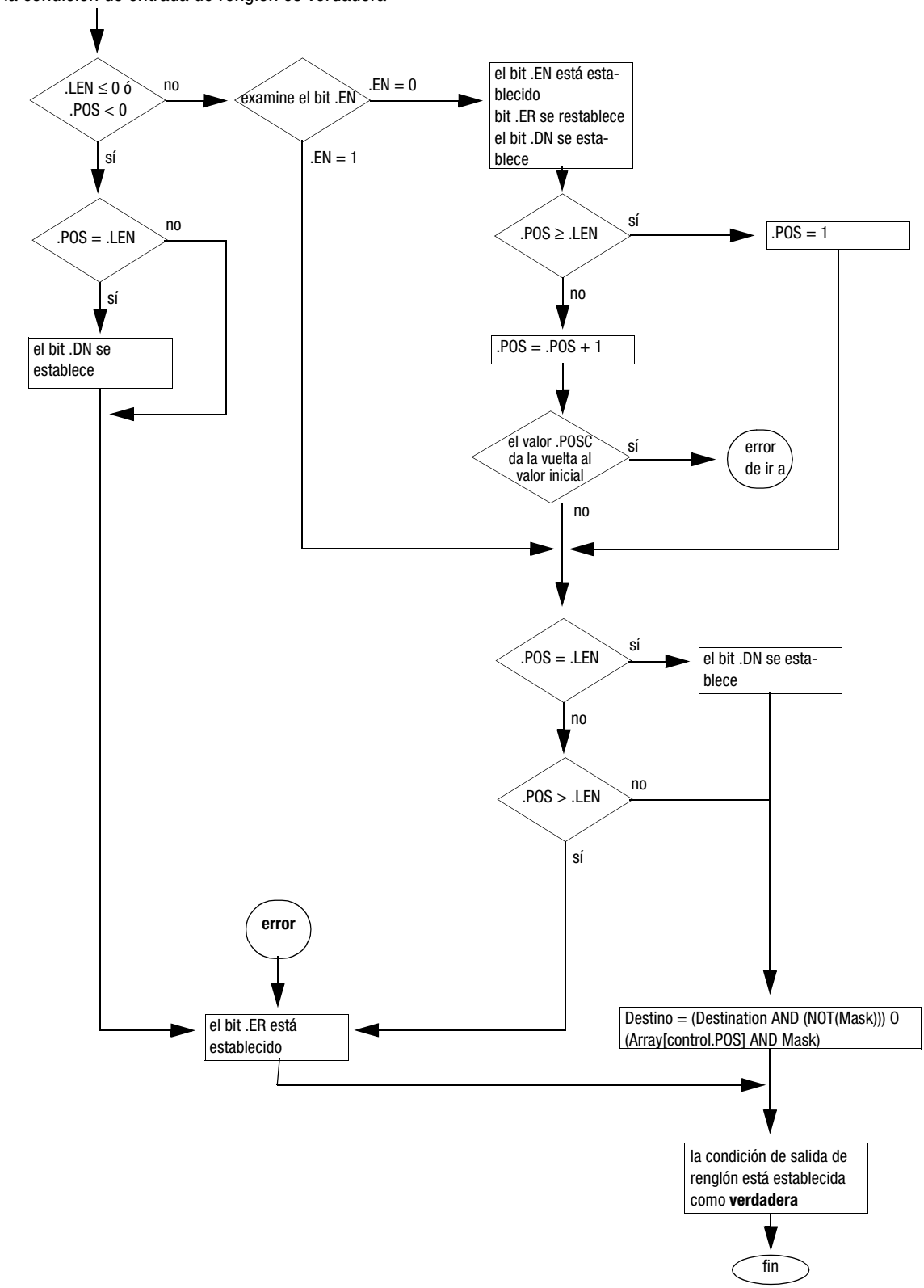
Prefijo:	Descripción:
16#	hexadecimal por ejemplo; 16#0F0F
8#	octal por ejemplo; 8#16
2#	binario por ejemplo; 2#00110011

Ejecución:

Condición:	Acción:
preescán	El bit .EN se establece para impedir una carga falsa cuando se inicia el escán del programa. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit .EN se restablece. La condición de salida de renglón se establece como falsa.

Condición: **Acción:**

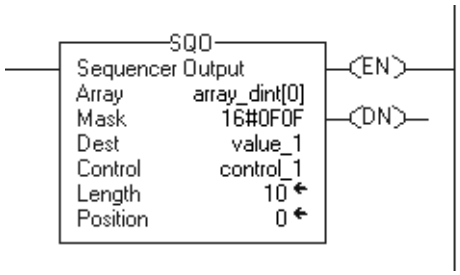
la condición de entrada de renglón es verdadera



Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de SQO:



Una vez habilitada, la instrucción SQO incrementa la posición, pasa los datos en dicha posición a *array_dint* a través de la máscara y almacena el resultado en *value_1*.

Operando SQO:	**Ejemplos de valores (usando DINT mostrados en la pantalla en formato binario):**
Arreglo	`xxxxxxxx xxxxxxxx xxxx0101 xxxx1010`
Máscara	`00000000 00000000 00001111 00001111`
Destino	`xxxxxxxx xxxxxxxx xxxx0101 xxxx1010`

Una cifra 0 en la máscara no cambia el bit (designado por xxxx en este ejemplo).

Cómo usar SQI con SQO

Si usted combina una instrucción SQI con una instrucción SQO, asegúrese de que ambas instrucciones usen los mismos valores de control, longitud y posición.

Cómo restablecer la posición de SQO

Cada vez que el controlador va del modo de programación al modo marcha, la instrucción SQO restablece (inicializa) el valor .POS. Para restablecer .POS en el valor de inicialización (.POS = 0), use la instrucción RES para borrar el valor de posición. Este ejemplo usa el estado del bit de primer escán para borrar el valor .POS.

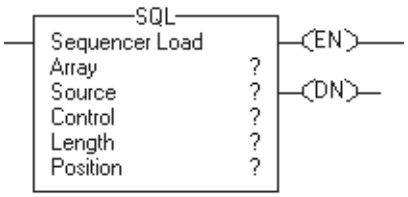
Otros formatos:

Formato:	Sintaxis:
texto neutro	`SQO(array,mask,destination,control,length,position);`
texto ASCII	`SQO array mask destination control length position`

Instrucciones relacionadas: SQI, SQL

Sequencer Load (SQL)

La instrucción SQL es una instrucción de salida.



Operandos:

Operando:	Tipo:	Formato:	Descripción:
Arreglo	DINT	tag de arreglo	arreglo de secuenciador especifique el primer elemento del arreglo de secuenciador **no** use CONTROL.POS en el subíndice
Origen	SINT INT **DINT**	tag valor inmediato	datos de entrada que se cargan en el arreglo de secuenciador
	Un tag SINT o INT se convierte a un valor DINT usando la extensión con signo. Consulte “Conversiones de datos” en la página A-6.		
Control	CONTROL	tag	estructura de control para la operación generalmente se usa el mismo CONTROL usado con las instrucciones SQI y SQO
Longitud	DINT	valor inmediato	número de elementos en el arreglo (tabla de secuenciadores) que se cargan
Posición	DINT	valor inmediato	posición actual en el arreglo el valor inicial es típicamente 0

Estructura CONTROL:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción SQL está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando todos los elementos especificados se han cargado en el arreglo.
.ER	BOOL	Se establece el bit de error cuando .LEN ≤ 0, .POS < 0 ó .POS $>$ .LEN.
.LEN	DINT	La longitud especifica el número de pasos en el arreglo de secuenciador.
.POS	DINT	La posición identifica el elemento que el controlador manipula actualmente.

Descripción: La instrucción SQL carga condiciones de referencia en un arreglo de secuenciador. Una vez habilitada, la instrucción SQL se incrementa a la próxima posición en el arreglo de secuenciador y carga el valor de origen en dicha posición. Si el bit .DN está establecido o si .POS $\geq$ .LEN, la instrucción establece .POS=1.

Generalmente se usa la misma estructura CONTROL usada con las instrucciones SQI y SQO.

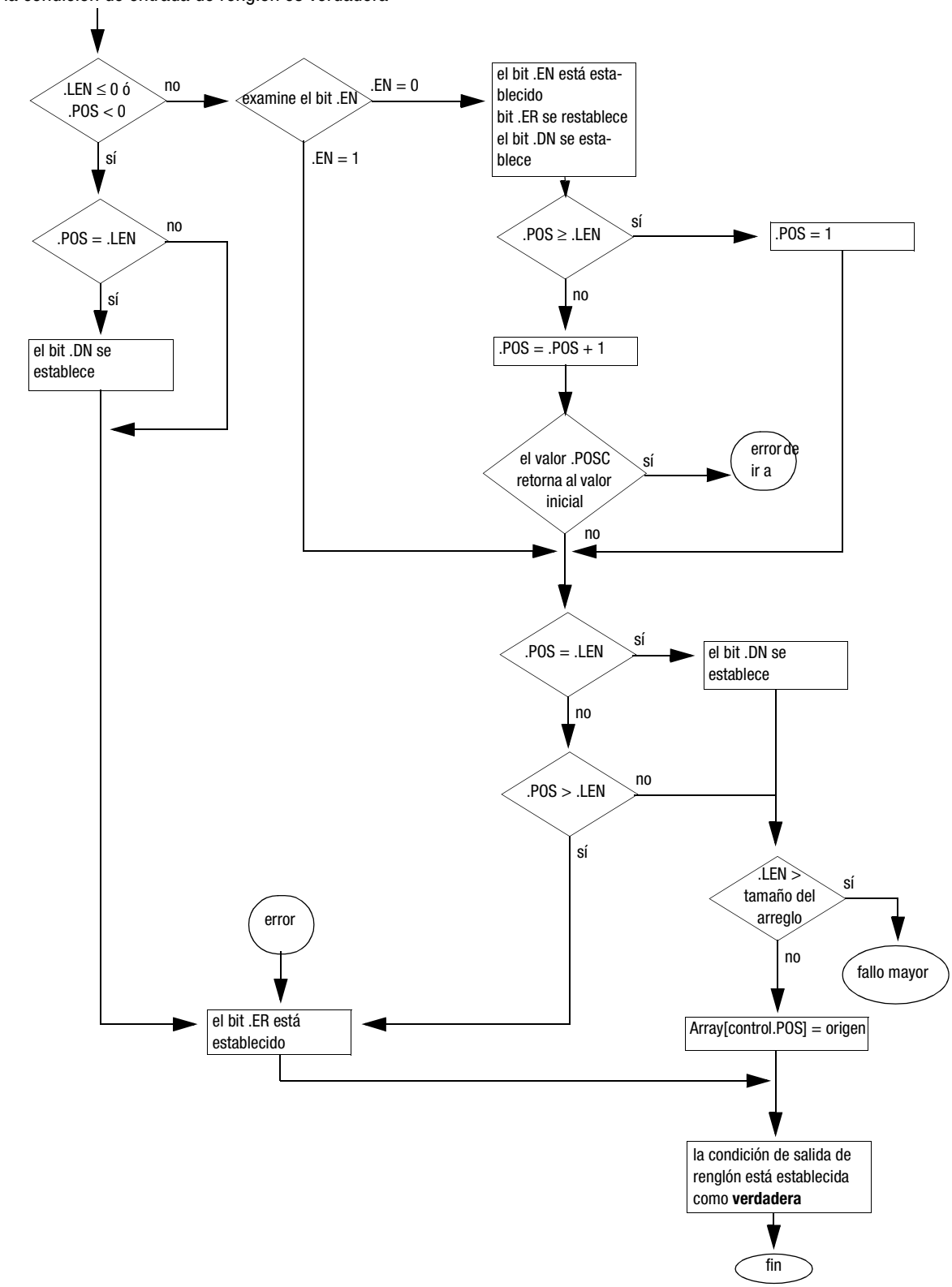
La instrucción SQL realiza la operación en la memoria de datos contiguos. Vea la página B-4, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información.

Ejecución:

Condición:	Acción:
preescán	El bit .EN se establece para impedir una carga falsa cuando se inicia el escán del programa. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	El bit .EN se restablece. La condición de salida de renglón se establece como falsa.

Condición: **Acción:**

la condición de entrada de renglón es verdadera

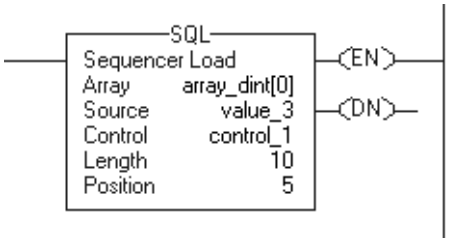


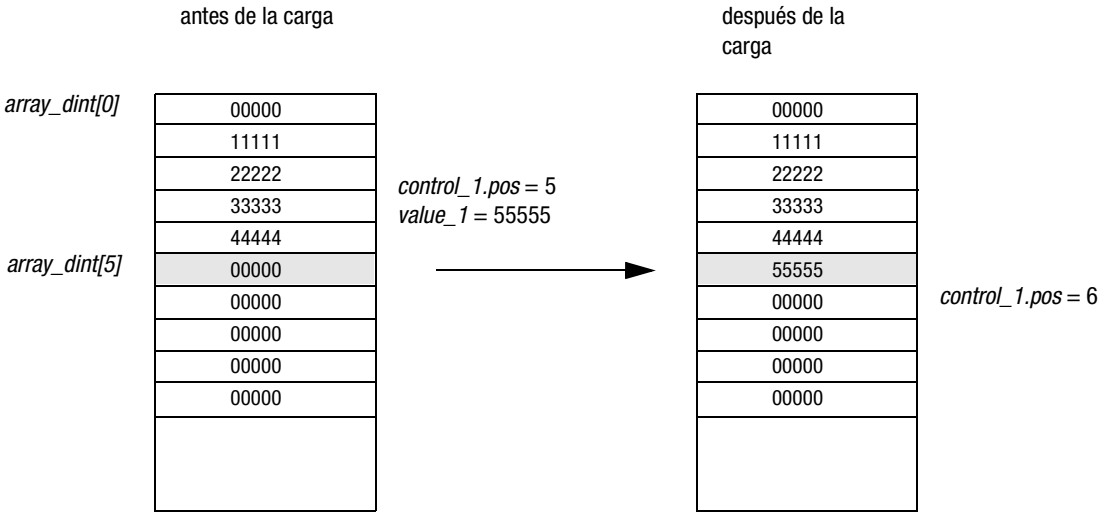
Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
Longitud > tamaño del arreglo	4	20

Ejemplo de SQL:





Una vez habilitada, la instrucción SQL carga *value_3* en la próxima posición en el arreglo de secuenciador, el cual es *array_dint[5]* en este ejemplo.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`SQL(array,source,control,length,position);`
texto ASCII	`SQL array source control length position`

Instrucciones relacionadas: SQI, SQO

Instrucciones de control de programa
(JMP, LBL, JSR, RET, SBR, TND, MCR, UID, UIE, AFI, NOP)

Introducción

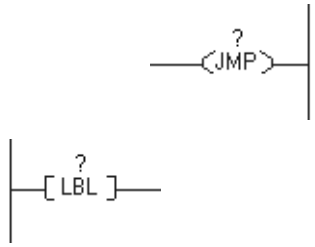
Use las instrucciones de control del programa para cambiar el flujo de la lógica.

Si usted desea:	Use esta instrucción:	Vea la página:
Saltar por encima de una sección de lógica que no necesita ejecutarse cada vez.	JMP LBL	10-2
Saltar a una rutina separada, pasar datos a la rutina, ejecutar la rutina y devolver los resultados.	JSR SBR RET	10-4
Marcar un fin temporal que detiene la ejecución de la rutina.	TND	10-10
Inhabilitar todos los renglones en una sección de lógica.	MCR	10-11
Inhabilitar las tareas del usuario.	UID	10-13
Habilitar las tareas del usuario.	UIE	10-14
Inhabilitar un renglón.	AFI	10-15
Insertar un marcador de posición en la lógica.	NOP	10-16

Jump to Label (JMP) Label (LBL)

La instrucción JMP es una instrucción de salida.
La instrucción LBL es una instrucción de entrada.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Instrucción JMP			
Nombre de etiqueta		nombre de etiqueta	introduzca un nombre para la instrucción LBL asociada
Instrucción LBL			
Nombre de etiqueta		nombre de etiqueta	la ejecución salta a la instrucción LBL con el nombre de etiqueta al cual se hizo referencia

Descripción: Las instrucciones JMP y LBL saltan porciones de la lógica de escalera. Una vez habilitada, la instrucción JMP salta a la instrucción LBL a la cual se hizo referencia y el controlador continúa ejecutándose en ese lugar. Cuando está inhabilitada, la instrucción JMP no afecta la ejecución de la lógica de escalera.

La instrucción JMP puede avanzar o invertir la ejecución de la lógica de escalera. El salto de avance a una etiqueta ahorra el tiempo de escán del programa omitiendo un segmento de la lógica hasta que sea necesario. El salto de retroceso permite que el controlador repita iteraciones de la lógica.

Tenga cuidado de no saltar hacia atrás un número de veces excesivo. El temporizador de control (watchdog) puede sobrepasar el tiempo de espera puesto que el controlador no llega nunca al fin de la lógica, lo cual, a su vez, hace que el controlador entre en fallo.

ATENCION: La lógica saltada no se escanea. Coloque la lógica crítica fuera de la zona saltada.

La instrucción LBL es el receptor de la instrucción JMP que tiene el mismo nombre de etiqueta. **Asegúrese de que la instrucción LBL sea la primera instrucción en el renglón.**

Un nombre de etiqueta debe ser único dentro de una rutina. El nombre puede:
- tener hasta 40 caracteres
- contener letras, números y caracteres de subrayado (_)

Ejecución:

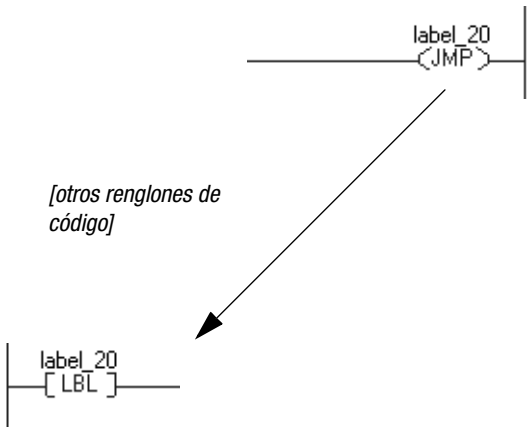
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera. La ejecución salta al renglón que contenga la instrucción LBL con el nombre de etiqueta al cual se hizo referencia.

Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
la etiqueta no existe	4	42

Ejemplo de JMP/LBL:



Una vez habilitada la instrucción JMP, la ejecución salta renglones sucesivos de lógica hasta llegar en el renglón que contiene la instrucción LBL con *label_20*.

Otros formatos:

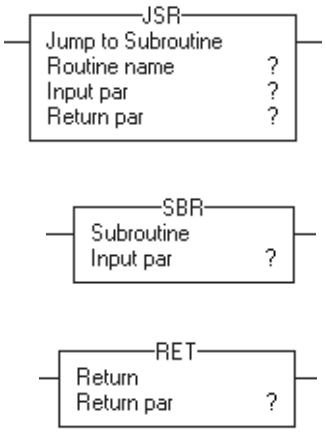
Formato:	Sintaxis:
texto neutro	`JMP(label_name);` `LBL(label_name);`
texto ASCII	`JMP label_name` `LBL label_name`

Instrucciones relacionadas: JSR, SBR, RET, FOR, BRK

Jump to Subroutine (JSR) Subroutine (SBR) Return (RET)

La instrucción JSR es una instrucción de salida.
La instrucción SBR es una instrucción de entrada.
La instrucción RET es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Instrucción JSR			
Nombre de rutina	rutina	nombre	rutina que se ejecuta
parámetro de entrada	SINT INT DINT REAL estructura	valor inmediato tag tag de arreglo	parámetros que pasan a la rutina
Parámetro de retorno	SINT INT DINT REAL estructura	tag tag de arreglo	parámetros (0-n) que se reciben de la rutina
Instrucción SBR			
parámetro de entrada	SINT INT DINT REAL estructura	tag tag de arreglo	parámetros (0-n) recibidos de JSR
Instrucción RET			
Parámetro de retorno	SINT INT DINT REAL estructura	valor inmediato tag tag de arreglo	parámetros (0-n) que devuelven a JSR

ATENCION: Un parámetro de entrada y su parámetro de retorno correspondiente deben ser del mismo tipo de datos o pueden ocurrir datos inesperados o una operación peligrosa.

Descripción: Las instrucciones JSR, SBR y RET conducen la ejecución de la lógica a una subrutina distinta dentro del programa, escanean dicha subrutina una vez y retornan al punto inicial.

También puede usar la instrucción JSR para probar una rutina de fallo o desactivar el controlador. Vea el documento *Logix5000 Controllers User Manual*, publicación 1756-6.5.12.

Una vez habilitada, la instrucción JSR conduce la ejecución de la lógica a la subrutina especificada y, si fuese necesario, pasa los parámetros a la subrutina. Si no hay parámetros de entrada, el control pasa de la instrucción JSR al primer renglón de la subrutina.

Una vez habilitada, la instrucción JSR pasa sus parámetros de entrada, si los hay, y conduce la ejecución al primer renglón de la subrutina. La instrucción SBR recibe los parámetros de entrada y copia los valores de los mismos en los tags especificados. El número y tipo de parámetros de entrada en la instrucción JSR debe coincidir con los de la instrucción SBR. Si la instrucción JSR tiene menos parámetros que la instrucción SBR asociada, el controlador entra en un fallo mayor. La instrucción JSR puede tener más parámetros de entrada que la instrucción SBR asociada sin causar un fallo.

Use la instrucción SBR en la subrutina solamente si desea pasar parámetros a la subrutina. Si usa una instrucción SBR, ésta debe ser la primera instrucción en el primer renglón de la rutina. La instrucción SBR opcional identifica los tags que almacenan los parámetros de entrada. Puede introducir una instrucción SBR sin parámetros para ayudar a identificar la rutina como subrutina.

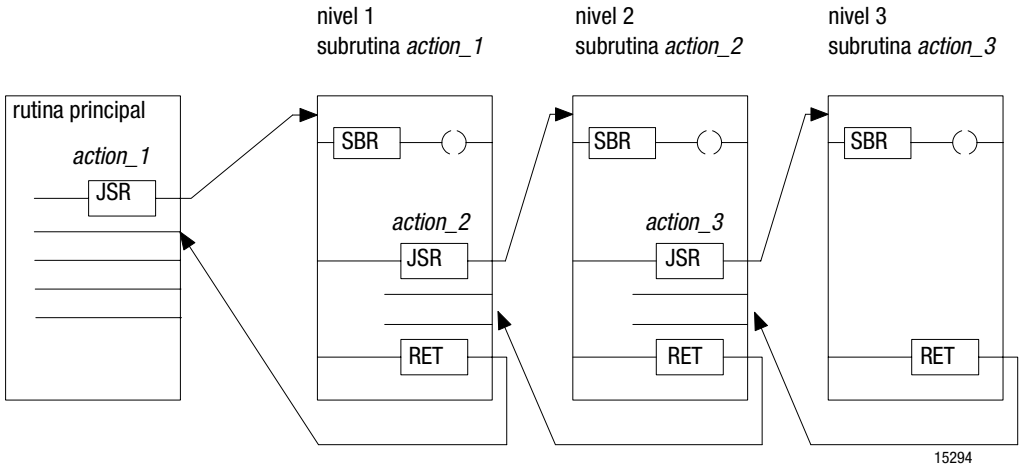
No puede colocar una instrucción SBR en una rutina principal.

Use la instrucción RET solamente si retorna parámetros a la instrucción JSR. La instrucción RET pone fin a la subrutina y, si fuese necesario, retorna los parámetros a la instrucción JSR. Una subrutina puede tener más de una instrucción RET.

Una vez habilitada, la instrucción RET pasa sus parámetros, si los hay, y reanuda la ejecución en el renglón que sigue a la instrucción JSR asociada. El número y tipo de parámetros de retorno en la instrucción RET debe coincidir con los de la instrucción SBR. Si el número de parámetros de retorno en la instrucción RET es menor que el número de parámetros de retorno en la instrucción JSR, el controlador entra en un fallo mayor. La instrucción RET puede tener más parámetros de retorno que la instrucción JSR asociada sin causar un fallo.

Cuando está inhabilitada, la instrucción RET no afecta la ejecución de la lógica. El controlador continúa ejecutando la subrutina actual.

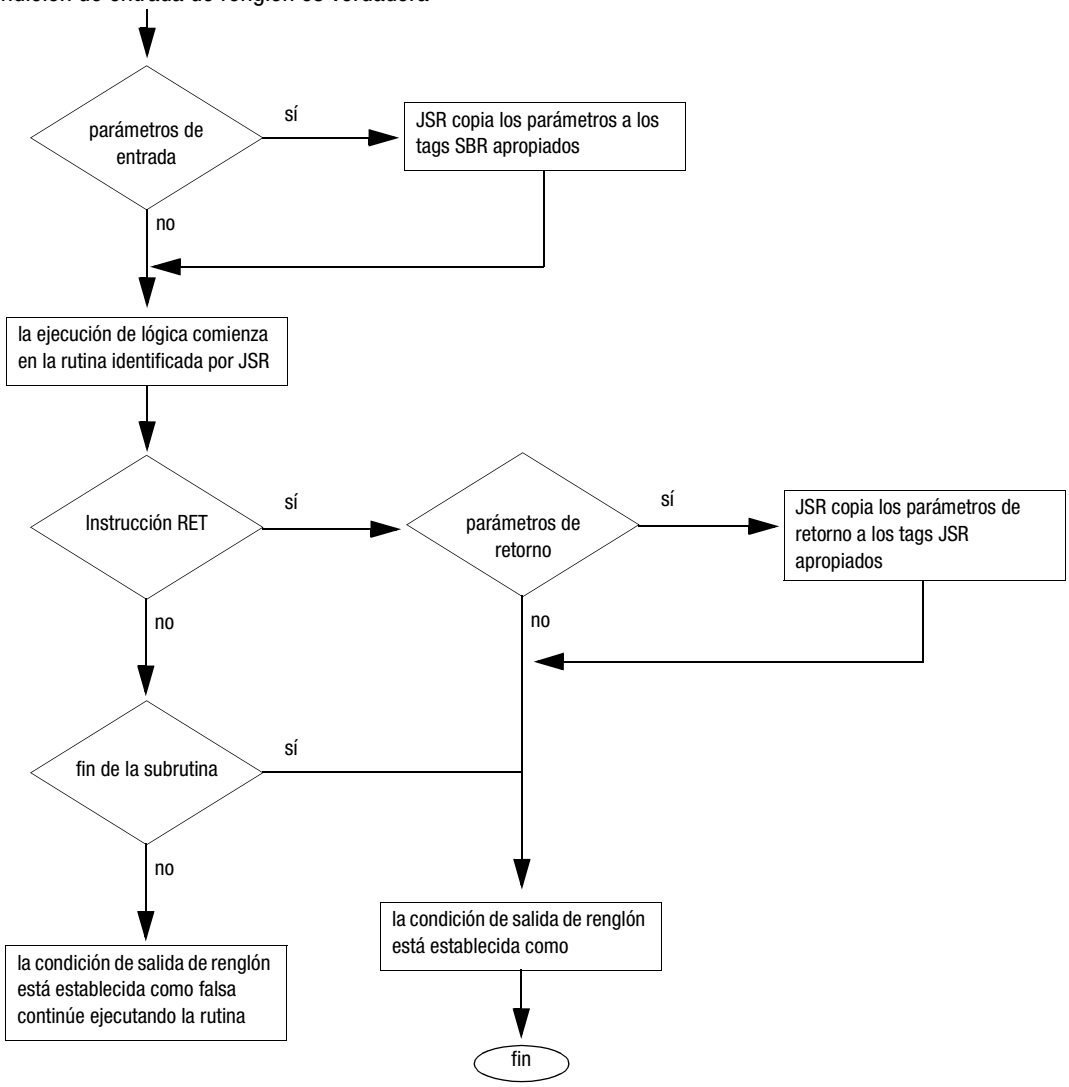
No hay restricciones, excepto la memoria del controlador, para el número de rutinas anidadas que pueden existir ni para el número de parámetros que se pasan o retornan.



Las instrucciones JSR, SBR y RET pasan parámetros hacia y desde una rutina según el valor. Esto significa que las instrucciones usan tiempo de ejecución y memoria adicionales para copiar los valores. Usted puede reducir el tiempo de ejecución obteniendo acceso directo a los datos del programa y del controlador cubiertos desde la rutina en vez de pasar los valores.

Puede pasar elementos de arreglo individuales, arreglos enteros, elementos de estructuras individuales y estructuras enteras. Los arreglos y las estructuras se copian de la misma manera que una instrucción COP copia los valores. Se recomienda pasar los parámetros de arreglo o estructura a parámetros de arreglo o estructura del mismo tipo, respectivamente.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa. El controlador ejecuta todas las subrutinas independientemente de la condición del renglón, pero ignora las instrucciones RET. Todos los parámetros de entrada pasan a la subrutina. Se pasan todos los parámetros de retorno, pero la instrucción RET no sale de la subrutina. Esto asegura que todos los renglones en la subrutina se preescanean. Si existen llamadas recursivas a la misma subrutina, la subrutina se preescanea solamente la primera vez que se llama. Si existen llamadas múltiples (no recursivas) a la misma subrutina, la subrutina se preescanea cada vez.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Indicadores de estado aritmético: no afectados

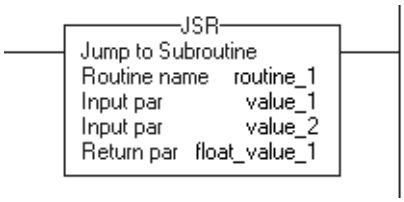
Condiciones de fallo:

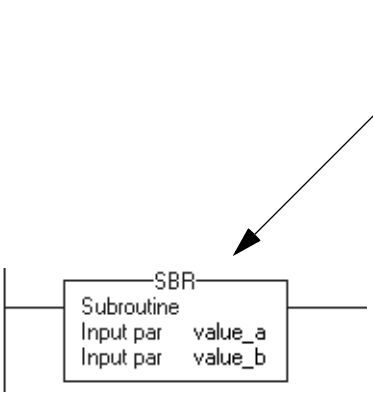
Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
la instrucción JSR tiene menos parámetros de entrada que la instrucción SBR	4	31
la instrucción JSR salta a una rutina de fallo	4 ó proporcionado por el usuario	0 ó proporcionado por el usuario
la instrucción RET tiene menos parámetros de retorno que la instrucción JSR	4	31

Ejemplo de JSR/SBR/RET:

ejemplo 1

Una vez habilitada, la instrucción JSR pasa *value_1* y *value_2* a *routine_1*.

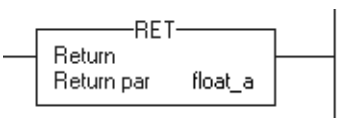




La instrucción SBR recibe *value_1* y *value_2* desde la instrucción JSR y copia estos valores a *value_a* y *value_b*, respectivamente. La ejecución de la lógica continúa en esta rutina.

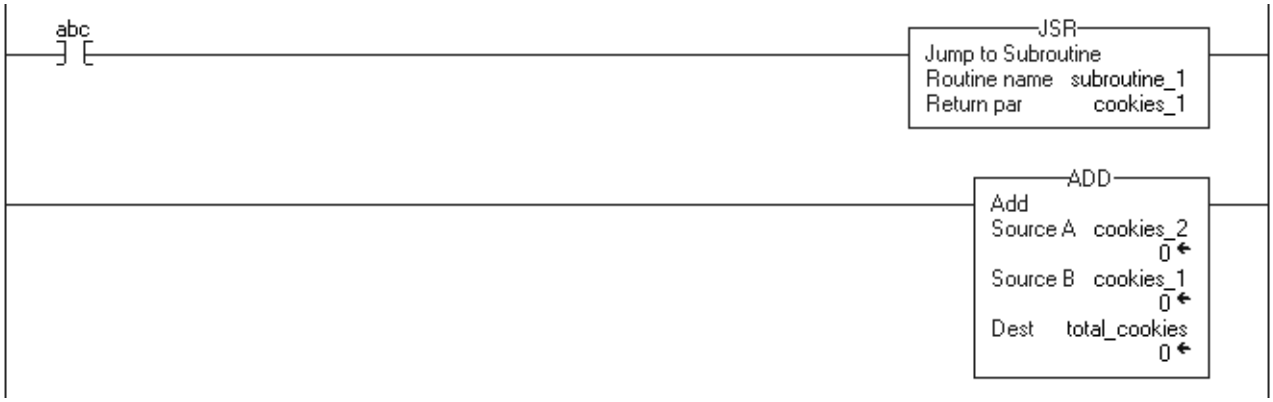
[otros renglones de código]

Una vez habilitada, la instrucción RET envía *float_a* a la instrucción JSR. La instrucción JSR recibe *float_a* y copia el valor a *float_value_1*. La ejecución de lógica continúa con la próxima instrucción que sigue a la instrucción JSR.

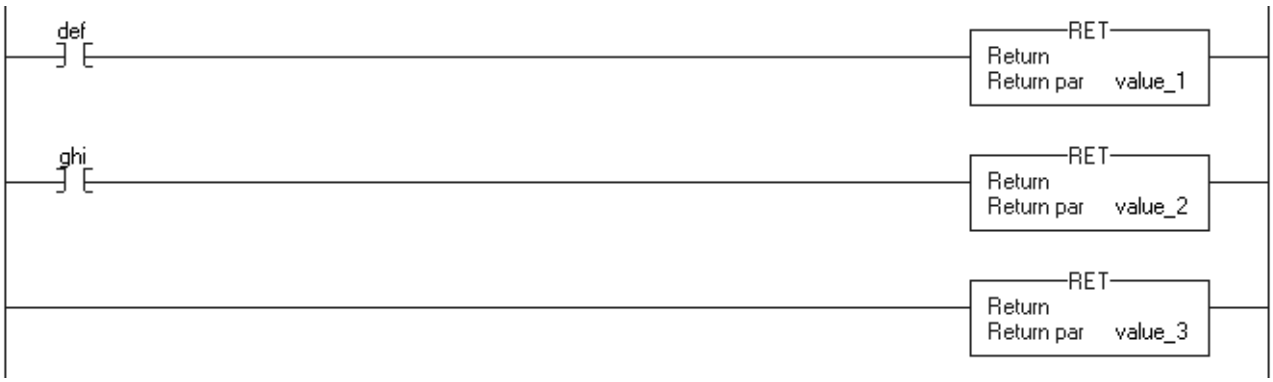


ejemplo 2

rutina principal



rutina *subroutine_1*



Si *abc* se hace verdadero, se habilita la instrucción JSR y el control pasa a *subroutine_1*. Si *def* se habilita, la instrucción RET retorna *value_1* al parámetro JSR *cookies_1* y el resto de la subrutina no se escanea. Si *ghi* se habilita, la instrucción RET retorna *value_2* al parámetro JSR *cookies_1* y el resto de la subrutina no se escanea. Si no se habilitan *def* ni *ghi*, la instrucción RET retorna *value_3* al parámetro JSR *cookies_1*. Luego la instrucción ADD suma el valor de *cookies_1* a *cookies_2* y almacena el resultado en *total_cookies*.

Otros formatos:

Formato:	Sintaxis:

texto neutro	`JSR(routine_name,input_1,...input_n,return_1,..return_n);`
	`SBR(routine_name,input_1,...input_n);`
	`RET(return_1,...return_n);`

texto ASCII	`JSR routine_name input_1 ... input_n return_1 ... return_n`
	`SBR routine_name input_1 ... input_n`
	`RET return_1 ... return_n`

Instrucciones relacionadas: JMP, LBL, FOR, BRK

Temporary End (TND)

La instrucción TND es una instrucción de salida.

Operandos: Ninguno

Descripción: La instrucción TND sirve como límite.

Una vez habilitada, la instrucción TND permite que el controlador ejecute la lógica solamente hasta esta instrucción.

Una vez habilitada, la instrucción TND sirve como el fin de la rutina. Cuando el controlador escanea una instrucción TND, el controlador se mueve al fin de la rutina actual. Si la instrucción TND es una subrutina, el control retorna a la rutina de llamada. Si la instrucción TND se encuentra en una rutina principal, el control retorna al próximo programa dentro de la tarea actual.

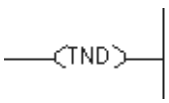
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera. Se interrumpe la rutina actual.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de TND: Usted puede usar la instrucción TND al depurar o resolver problemas para ejecutar la lógica hasta cierto punto. Mueva progresivamente la instrucción TND a través de la lógica a la vez que depura cada nueva sección.



Una vez habilitada la instrucción TND, el controlador ya no escanea la rutina actual.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`TND( );`
texto ASCII	`TND`

Instrucciones relacionadas: AFI, MCR, NOP

Master Control Reset (MCR)

La instrucción MCR es una instrucción de salida.

Operandos: Ninguno

—(MCR)—

Descripción: La instrucción MCR, usada en parejas, crea una zona de programa que puede inhabilitar todos los renglones dentro de las instrucciones MCR.

Una vez habilitada la zona MCR, los renglones en la zona MCR se escanean para ver si hay condiciones normales verdaderas o falsas. Cuando está inhabilitado, el controlador todavía escanea los renglones dentro de una zona MCR, pero se reduce el tiempo del escán puesto que están inhabilitadas las salidas no retentivas en la zona. La condición de entrada de renglón es falsa para todas las instrucciones dentro de la zona MCR inhabilitada.

Cuando programe una zona MCR, recuerde que:
- Debe finalizar la zona con una instrucción MCR sin condiciones.
- No puede anidar una zona MCR dentro de otra.
- No salte en una zona MCR. Si la zona es falsa, el saltar a la zona activa la zona a partir del punto al cual se saltó hasta el fin de la zona.
- Si una zona MCR continúa hasta el fin de la rutina, no es necesario programar una instrucción MCR para finalizar la zona.

Importante: La instrucción MCR no reemplaza a un relé de control maestro cableado que proporciona la capacidad de paro de emergencia. Debe instalar un relé de control maestro cableado para proporcionar interrupción de emergencia de la alimentación eléctrica a las E/S.

ATENCION: No sobreponga ni anide las zonas MCR. Cada zona debe ser separada y completa. Si se sobreponen o se anidan, puede ocurrir una operación inesperada de la máquina con posibles daños del equipo o lesiones personales.

Coloque las operaciones críticas fuera de la zona MCR. Si inicia instrucciones tales como los temporizadores en una zona MCR, se detiene la ejecución de la instrucción cuando la zona se inhabilita y el temporizador se restablece.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa. Se escanean las instrucciones en la zona, pero la condición de entrada de renglón es falsa y las salidas no retentivas en la zona se inhabilitan.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera. Las instrucciones en la zona se escanean normalmente.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de MCR:

Cuando se habilita la primera instrucción MCR (*input_1*, *input_2* y *input_3* están establecidos), el controlador ejecuta los renglones en la zona MCR (entre las dos instrucciones MCR) y establece o restablece las salidas según las condiciones de entrada.
Cuando se inhabilita la primera instrucción MCR (*input_1*, *input_2* y *input_3* no están todos establecidos), el controlador ejecuta los renglones en la zona MCR (entre las dos instrucciones MCR) y la condición de entrada de renglón se hace falsa para todos los renglones en la zona MCR independientemente de las condiciones de entrada.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`MCR( );`
texto ASCII	`MCR`

Instrucciones relacionadas: AFI, NOP, TND

User Interrupt Disable (UID)

La instrucción UID es una instrucción de salida.

Operandos: Ninguno

Descripción: La instrucción UID inhabilita temporalmente la conmutación entre tareas del usuario.

Una vez habilitada la instrucción UID, la tarea actual continúa ejecutándose sin interrupción por parte de las tareas de alta prioridad hasta que se ejecuta una instrucción UIE o se llega al fin del programa. La instrucción UID no inhabilita la ejecución de una rutina de fallo o tarea de fallo.

Una vez habilitada, la instrucción UID incrementa un contador interno. Siempre que este valor de contador no ea igual a cero, la tarea que se ejecuta actualmente no se puede interrumpir. Puede anidar hasta 65,535 niveles de instrucciones UID.

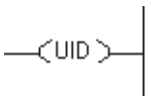
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Se evita la interrupción por tareas de alta prioridad. Se incrementa el contador interno UID. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo UID:



Una vez habilitada, la instrucción UID inhabilita temporalmente la conmutación entre tareas del usuario.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`UID();`
texto ASCII	`UID`

Instrucciones relacionadas: UIE

User Interrupt Enable (UIE)

La instrucción UIE es una instrucción de salida.

Operandos: ninguna

Descripción: La instrucción UIE vuelve a habilitar la conmutación las tareas del usuario.

Si la instrucción UIE está habilitada y el contador interno es mayor que cero, el contador se decrementa. Cuando el contador es igual a cero, la tarea que se ejecuta actualmente puede volver a interrumpirse. Las tareas de alta prioridad que no pudieron interrumpir la tarea actual ahora se ejecutan.

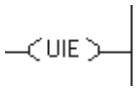
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	Se decrementa el contador interno UID. Si el conteo interno es igual a 0, las tareas de alta prioridad pueden interrumpir la tarea actual. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de UIE:



Una vez habilitada, la instrucción UIE vuelve a habilitar la conmutación entre tareas del usuario.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`UIE();`
texto ASCII	`UIE`

Instrucciones relacionadas: UID

Always False (AFI)

La instrucción AFI es una instrucción de entrada.

–[AFI]–

Operandos: Ninguno

Descripción: La instrucción AFI establece su condición de salida de renglón como falsa.

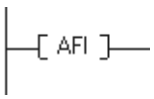
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón se establece como falsa.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: Ninguno

Ejemplo de AFI: Use la instrucción AFI para inhabilitar temporalmente un renglón mientras depura un programa.



Una vez habilitada, la instrucción AFI inhabilita todas las instrucciones en este renglón.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`AFI();`
texto ASCII	`AFI`

Instrucciones relacionadas: MCR, NOP, TND

No Operation (NOP)

La instrucción NOP es una instrucción de entrada y salida.

Operandos: Ninguno

Descripción: La instrucción NOP sirve como indicador de posición.

Usted puede colocar la instrucción NOP en cualquier parte de un renglón. Una vez habilitada, la instrucción NOP no realiza una operación. Una vez inhabilitada, la instrucción NOP no realiza una operación.

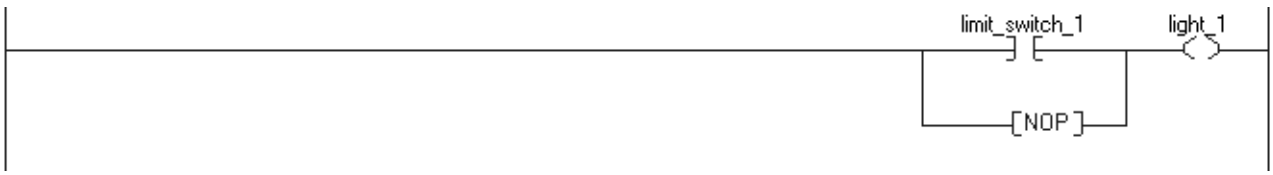
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: Ninguno

Ejemplo NOP: Esta instrucción es útil para localizar bifurcaciones no condicionales cuando coloca la instrucción NOP en la bifurcación.



La instrucción NOP omite la instrucción XIC para habilitar la salida.

Otros formatos:

Formato:	Sintaxis:
texto neutro	NOP ( ) ;
texto ASCII	NOP

Instrucciones relacionadas: AFI, MCR, TND

Instrucciones FOR/interrupción
(FOR, BRK, RET)

Introducción

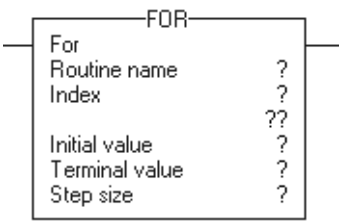
Use la instrucción FOR para llamar repetidamente una subrutina. Use la instrucción BRK para interrumpir la ejecución de una subrutina.

Si usted desea:	Use esta instrucción:	Vea la página:
Ejecutar repetidamente una rutina.	FOR	11-2
Interrumpir la ejecución repetida de una rutina.	BRK	11-5
Retornar a la instrucción FOR.	RET	11-6

For (FOR)

La instrucción FOR es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Nombre de rutina	ROUTINE	nombre de rutina	rutina que se ejecuta
Indice	DINT	tag	cuenta cuántas veces la rutina se ha ejecutado
Valor inicial	SINT INT DINT	valor inmediato tag	valor en que se comienza el índice
Valor terminal	SINT INT DINT	valor inmediato tag	valor en que se detiene la ejecución de la rutina
Tamaño de paso	SINT INT DINT	valor inmediato tag	cantidad que se añade al índice cada vez que la instrucción FOR ejecuta la rutina

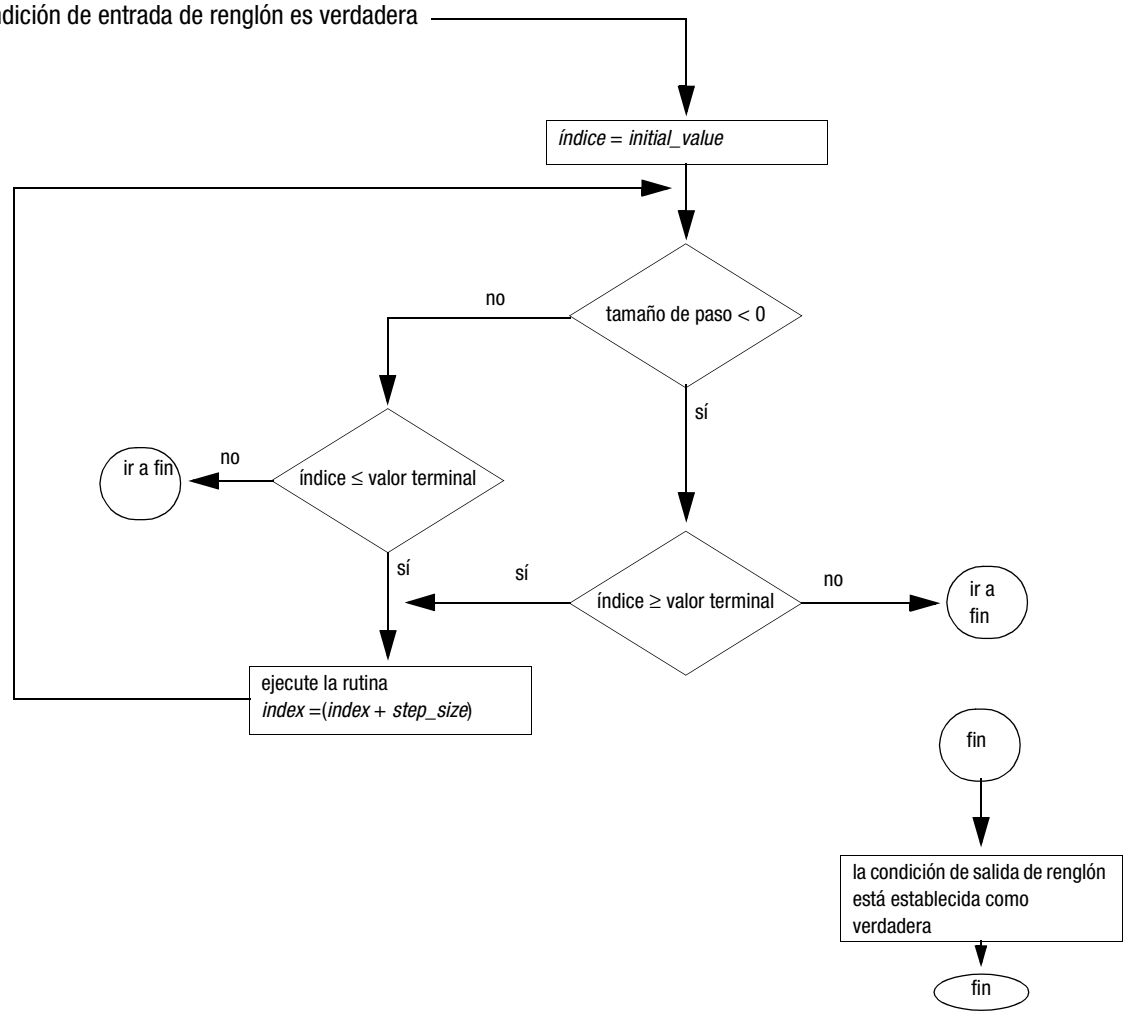
Descripción: La instrucción FOR ejecuta una rutina repeditamente.

Una vez habilitada, la instrucción FOR ejecuta repetidamente la rutina hasta que el valor del índice excede el valor terminal. Esta instrucción no pasa parámetros a la rutina.

Cada vez que la instrucción FOR ejecuta la rutina, añade el tamaño de paso al índice.

Tenga cuidado de no realizar demasiados lazos en un solo escán. Una cantidad excesiva de repeticiones puede causar que el temporizador de control (watchdog) del controlador sobrepase el tiempo de espera, lo cual causará un fallo mayor.

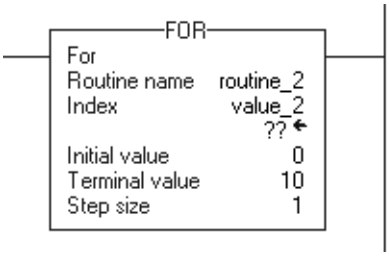
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa. El controlador ejecuta la subrutina una vez. Si existen instrucciones FOR recursivas en la misma subrutina, la subrutina se preescanea solamente la primera vez que se llama. Si existen múltiples instrucciones FOR (no recursivas) a la misma subrutina, la subrutina se preescanea cada vez.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de FOR:



Una vez habilitada, la instrucción FOR ejecuta repetidamente *routine_2* e incrementa *value_2* en un valor de 1 cada vez. Cuando *value_2* es > 10 o una instrucción BRK está habilitada, la instrucción FOR ya no ejecuta *routine_2*.

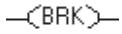
Otros formatos:

Formato:	Sintaxis:
texto neutro	`FOR(routine_name,index,initial_value,terminal_value,step_size);`
texto ASCII	`FOR routine_name index initial_value terminal_value step_size`

Instrucciones relacionadas: BRK, JMP, LBL, JSR, SBR, RET

Break (BRK)

La instrucción BRK es una instrucción de salida.



Operandos: ninguna

Descripción: La instrucción BRK interrumpe la ejecución de una rutina que fue llamada por una instrucción FOR.

Una vez habilitada, la instrucción BRK sale de la rutina y retorna el controlador a la instrucción que sigue a FOR.

Si hay instrucciones FOR anidadas, una instrucción BRK retorna el control a la instrucción FOR más profunda.

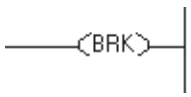
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera. La ejecución retorna a la instrucción que sigue a la instrucción FOR que hizo la llamada.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de BRK:



Una vez habilitada, la instrucción BRK ya no ejecuta la rutina actual y retorna a la instrucción que sigue a la instrucción FOR que hizo la llamada.

Otros formatos:

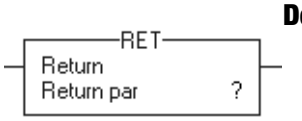
Formato:	Sintaxis:
texto neutro	`BRK( );`
texto ASCII	`BRK`

Instrucciones relacionadas: FOR, JMP, LBL, JSR, SBR, RET

Return (RET)

La instrucción RET es una instrucción de salida.

Operandos: ninguna



Descripción: La instrucción RET retorna a la instrucción FOR que hizo la llamada. La instrucción FOR no usa parámetros. La instrucción FOR ignora los parámetros que usted introduce en una instrucción RET.

Una vez habilitada, la instrucción RET retorna a la instrucción FOR. La instrucción FOR incrementa el valor del índice según el tamaño de paso y vuelve a ejecutar la subrutina. Si el valor del índice excede el valor terminal, la instrucción FOR concluye y la ejecución continúa a la instrucción que sigue a la instrucción FOR.

También puede usar una instrucción TND para finalizar la ejecución de una subrutina.

Ejecución:

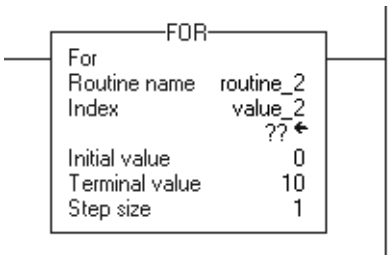
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

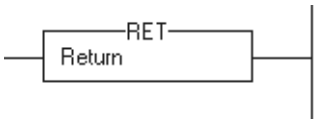
Ejemplo de RET:

rutina que hace



Una vez habilitada, la instrucción FOR ejecuta repetidamente *routine_2* e incrementa *value_2* en un valor de 1 cada vez. Cuando *value_2* es > 10 o una instrucción BRK está habilitada, la instrucción FOR ya no ejecuta *routine_2*.

subrutina



Una vez habilitada, la instrucción RET retorna a la instrucción FOR que hizo la llamada. La instrucción FOR vuelve a ejecutar la subrutina e incrementa el valor del índice según el tamaño de paso o, si el valor del índice excede el valor terminal, la instrucción FOR concluye y la ejecución continúa a la instrucción que sigue a la instrucción FOR.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`RET( );`
texto ASCII	RET

Instrucciones relacionadas: BRK, JMP, LBL, TND

Instrucciones especiales
(FBC, DDT, DTR, PID)

Introducción

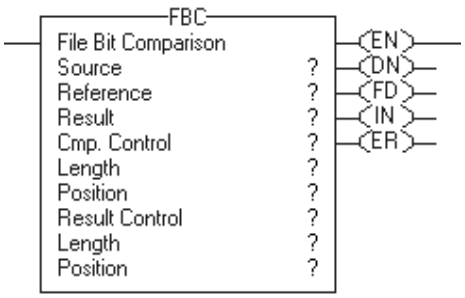
Las instrucciones especiales realizan operaciones para aplicaciones específicas.

Si usted desea:	Use esta instrucción:	Vea la página:
Comparar datos contra una referencia confiable conocida y registrar las desigualdades.	FBC	12-2
Comparar los datos contra una referencia confiable conocida, registrar las desigualdades y actualizar la referencia para que coincida con el origen.	DDT	12-9
Pasar los datos de origen mediante una máscara y comparar el resultado contra datos de referencia. Luego escribir el origen en la referencia para la próxima comparación.	DTR	12-16
Controlar un lazo PID.	PID	12-19

File Bit Comparison (FBC)

La instrucción FBC es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	DINT	tag de arreglo	arreglo que se compara contra la referencia **no** use CONTROL.POS en el subíndice
Referencia	DINT	tag de arreglo	arreglo que se compara contra el origen **no** use CONTROL.POS en el subíndice
Resultado:	DINT	tag de arreglo	arreglo que almacena el resultado **no** use CONTROL.POS en los subíndices
Control Cmp	control	estructura	estructura de control para la comparación
Longitud	DINT	valor inmediato	número de bits que se comparan
Posición	DINT	valor inmediato	posición actual en el origen el valor inicial es típicamente 0
Control de resultado	control	estructura	estructura de control para los resultados
Longitud	DINT	valor inmediato	número de ubicaciones de almacenamiento en el resultado
Posición	DINT	valor inmediato	posición actual en el resultado el valor inicial es típicamente 0

ATENCION: Use tags diferentes para la estructura de control de comparación y la estructura de control de resultado. El uso del mismo tag para ambas estructuras puede resultar en una operación inesperada, la cual puede causar posibles daños al equipo y/o lesiones personales.

Estructura CONTROL de comparación:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción FBC está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando la instrucción FBC compara el último bit en los arreglos de origen y referencia.
.FD	BOOL	Se establece el bit de encontrado cada vez que la instrucción FBC registra una desigualdad (operación uno a uno) o después de registrar todas las desigualdades (operación de todos por escán).
.IN	BOOL	El bit de inhibición indica el modo de búsqueda FBC. 0 = modo todos 1 = modo de una desigualdad a la vez
.ER	BOOL	Se establece el bit de error si la comparación .POS < 0, la comparación .LEN < 0, el resultado .POS < 0 ó el resultado .LEN < 0. La instrucción ya no se ejecuta hasta que el programa restablece el bit .ER.
.LEN	DINT	El valor de longitud identifica el número de bits que se comparan.
.POS	DINT	El valor de posición identifica el bit actual.

Estructura CONTROL de resultado:

Mnemónico:	Tipo de datos:	Descripción:
.DN	BOOL	Se establece el bit de efectuado cuando el arreglo de resultados está lleno.
.LEN	DINT	El valor de longitud identifica el número de ubicaciones de almacenamiento en el arreglo de resultados.
.POS	DINT	El valor de posición identifica la posición actual en el arreglo de resultados.

Descripción: La instrucción FBC compara los bits de un arreglo de origen contra los bits de un arreglo de referencia.

Una vez habilitada, la instrucción FBC compara los bits del arreglo de origen contra los bits del arreglo de referencia y registra el número de bit de cada desigualdad en el arreglo de resultados.

La instrucción FBC opera en la memoria de datos contiguos. Vea la página, *Cómo ver un arreglo como un bloque de memoria,* para obtener más información B-4.

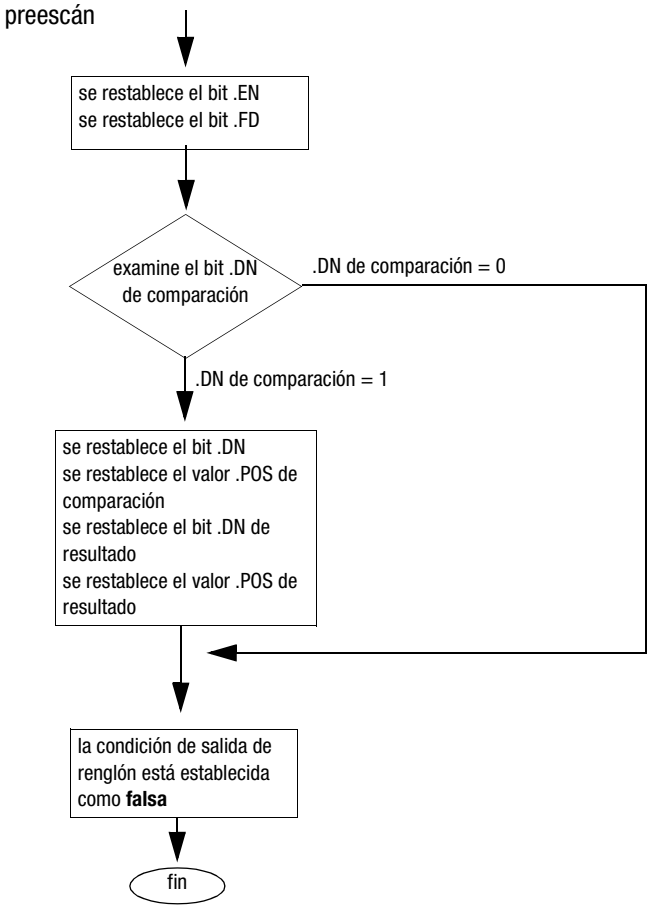
La diferencia entre las instrucciones DDT y FBC es que cada vez que la instrucción DDT encuentra una desigualdad, la instrucción cambia el bit de referencia para que coincida con el bit de origen. La instrucción FBC no cambia el bit de referencia.

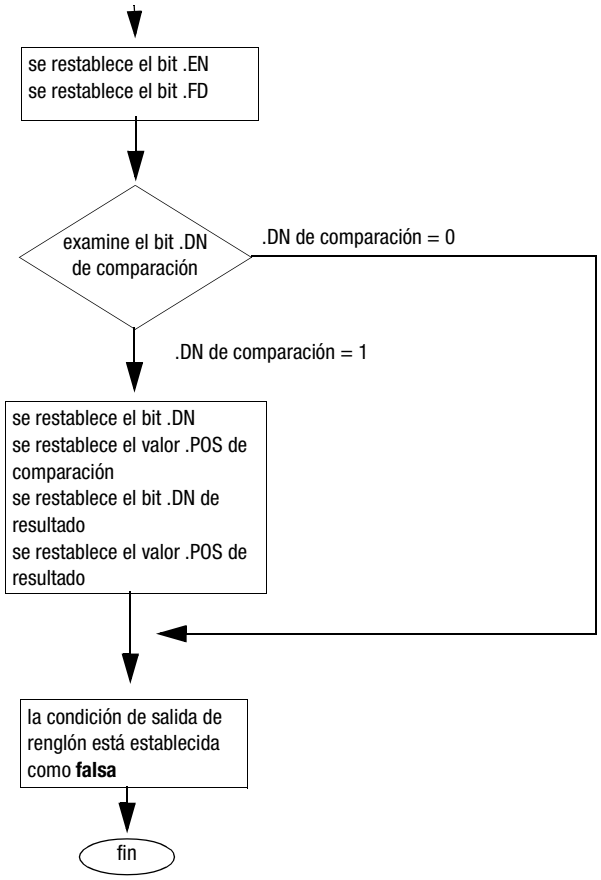
Cómo seleccionar el modo buscar

Si usted desea detectar:	Seleccione este modo:
Una desigualdad a la vez	Establezca el bit .IN en la estructura CONTROL de comparación. Cada vez que la condición de entrada de renglón va de falso a verdadero, la instrucción FBC busca la próxima desigualdad entre los arreglos de origen y referencia. Cuando la instrucción encuentra una desigualdad, ésta establece el bit .FD, registra la posición de la desigualdad y detiene la ejecución.
Todas las desigualdades	Restablezca el bit .IN en la estructura CONTROL de comparación. Cada vez que la condición de entrada de renglón va de falso a verdadero, la instrucción FBC busca todas las desigualdades entre los arreglos de origen y referencia.

Ejecución:

Condición: **Acción:**

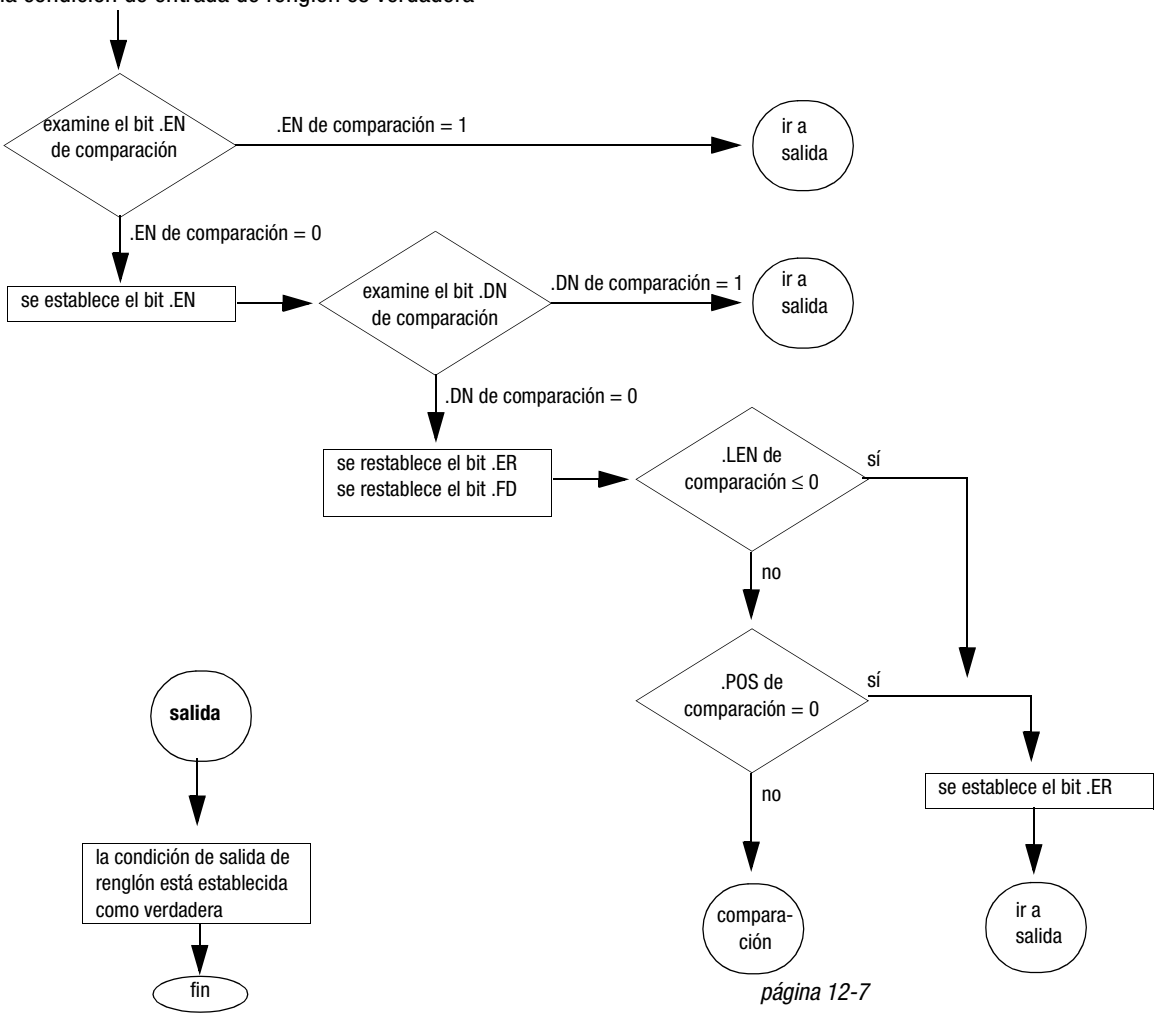

preescán
se restablece el bit .EN
se restablece el bit .FD
examine el bit .DN de comparación
.DN de comparación = 0
.DN de comparación = 1
se restablece el bit .DN
se restablece el valor .POS de comparación
se restablece el bit .DN de resultado
se restablece el valor .POS de resultado
la condición de salida de renglón está establecida como **falsa**
fin

Condición:	**Acción:**
la condición de entrada de renglón es falsa	

se restablece el bit .EN
se restablece el bit .FD
examine el bit .DN de comparación
.DN de comparación = 0
.DN de comparación = 1
se restablece el bit .DN
se restablece el valor .POS de comparación
se restablece el bit .DN de resultado
se restablece el valor .POS de resultado
la condición de salida de renglón está establecida como **falsa**
fin

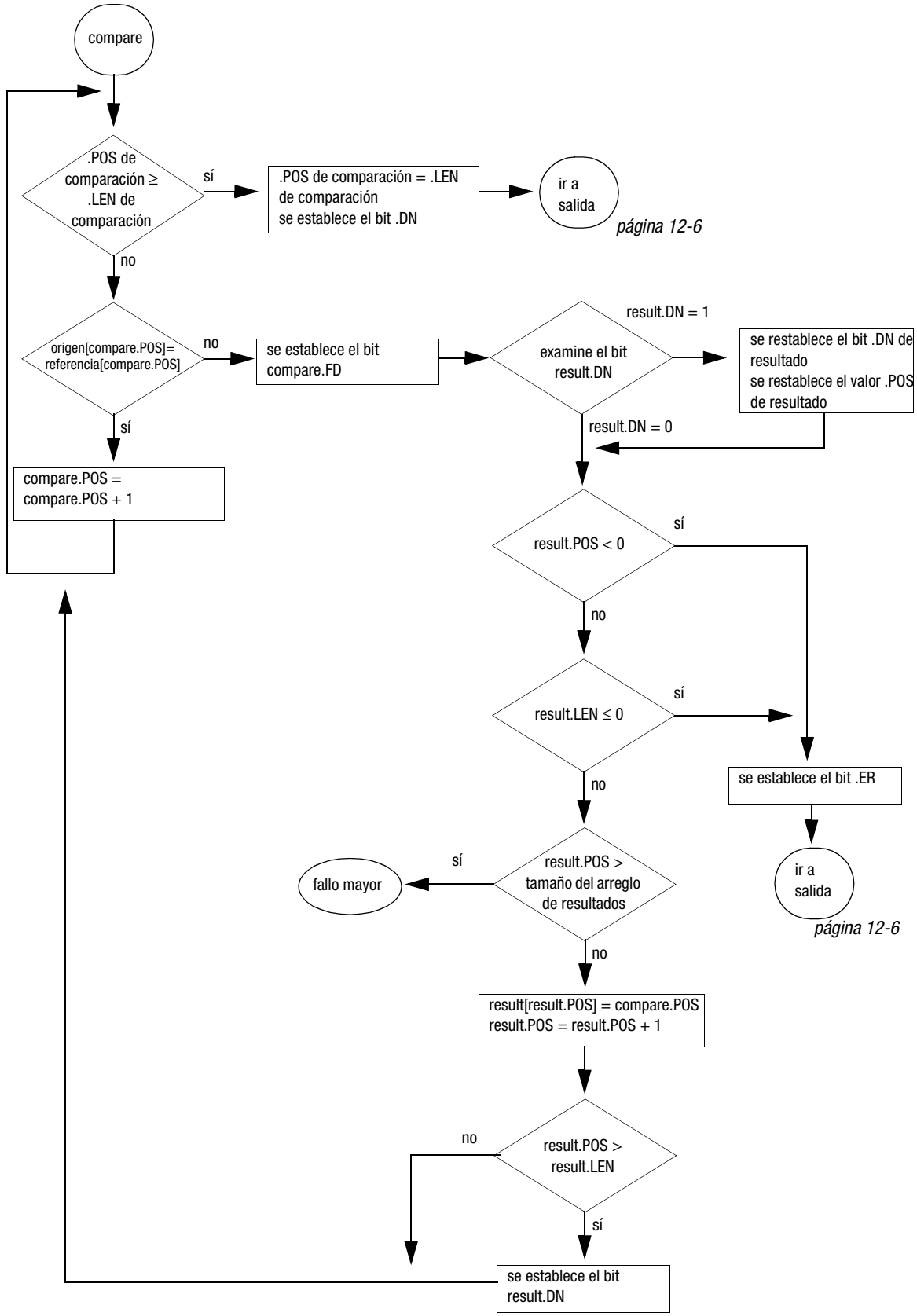
Condición:	**Acción:**

la condición de entrada de renglón es verdadera



página 12-7

Condición: **Acción:**

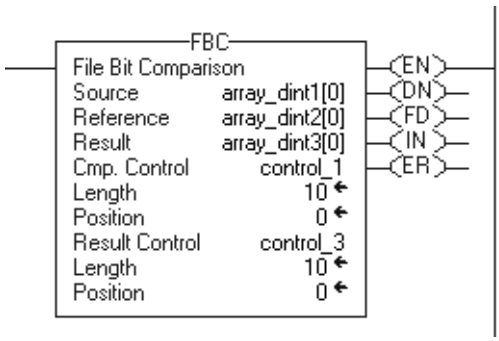


Indicadores de estado aritmético: no afectados

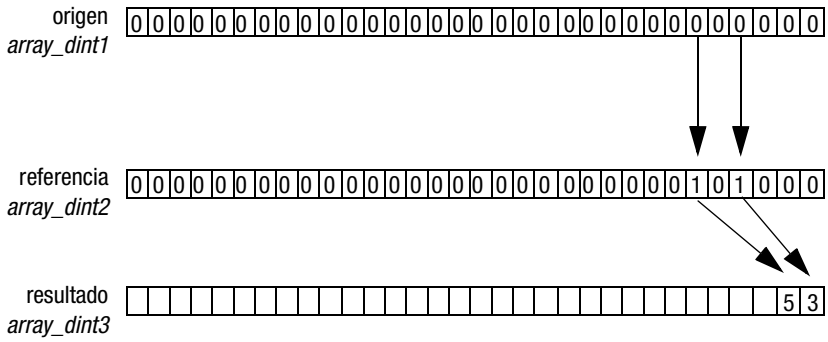
Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
Result.POS > tamaño del arreglo de resultados	4	20

Ejemplo de FBC:



Una vez habilitada, la instrucción FBC compara el origen *array_dint1* contra la referencia *array_dint2* y almacena la ubicaciones de las desigualdades en el resultado *array_dint3*.



Otros formatos:

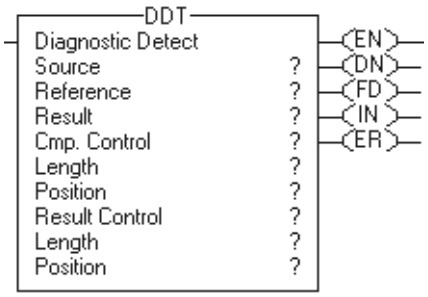
Formato:	Sintaxis:
texto neutro	FBC(*source,reference,result,cmp_control,length,position,result_control,length,position*);
texto ASCII	FBC *source reference result cmp_control length position result_control length position*

Instrucciones relacionadas: DDT, DTR

Diagnostic Detect (DDT)

La instrucción DDT es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	DINT	tag de arreglo	arreglo que se compara contra la referencia **no** use CONTROL.POS en el subíndice
Referencia	DINT	tag de arreglo	arreglo que se compara contra el origen **no** use CONTROL.POS en el subíndice
Resultado:	DINT	tag de arreglo	arreglo que almacena los resultados **no** use CONTROL.POS en el subíndice
Control Cmp	CONTROL	estructura	estructura de control para la comparación
Longitud	DINT	valor inmediato	número de bits que se comparan
Posición	DINT	valor inmediato	posición actual en el origen el valor inicial es típicamente 0
Control de resultado	control	estructura	estructura de control para los resultados
Longitud	DINT	valor inmediato	número de ubicaciones de almacenamiento en el resultado
Posición	DINT	valor inmediato	posición actual en el resultado el valor inicial es típicamente 0

ATENCION: Use tags diferentes para la estructura de control de comparación y la estructura de control de resultado. El uso del mismo tag para ambas estructuras puede resultar en una operación inesperada, la cual puede causar posibles daños al equipo y/o lesiones personales.

Estructura CONTROL de comparación:

Mnemónico:	Tipo de datos:	Descripción:
.EN	BOOL	El bit de habilitación indica que la instrucción DDT está habilitada.
.DN	BOOL	Se establece el bit de efectuado cuando la instrucción DDT compara el último bit en los arreglos de origen y referencia.
.FD	BOOL	Se establece el bit de encontrado cada vez que la instrucción DDT registra una desigualdad (operación uno a uno) o después de registrar todas las desigualdades (operación de todos por escán).
.IN	BOOL	El bit de inhibición indica el modo de búsqueda DDT. 0 = modo todos 1 = modo de una desigualdad a la vez
.ER	BOOL	Se establece el bit de error si la comparación .POS < 0, la comparación .LEN < 0, el resultado .POS < 0 ó el resultado .LEN < 0. La instrucción ya no se ejecuta hasta que el programa restablece el bit .ER.
.LEN	DINT	El valor de longitud identifica el número de bits que se comparan.
.POS	DINT	El valor de posición identifica el bit actual.

Estructura CONTROL de resultado:

Mnemónico:	Tipo de datos:	Descripción:
.DN	BOOL	Se establece el bit de efectuado cuando el arreglo de resultados está lleno.
.LEN	DINT	El valor de longitud identifica el número de ubicaciones de almacenamiento en el arreglo de resultados.
.POS	DINT	El valor de posición identifica la posición actual en el arreglo de resultados.

Descripción: La instrucción DDT compara los bits en un arreglo de origen contra los bits en un arreglo de referencia para determinar los cambios de estado.

Una vez habilitada, la instrucción DDT compara los bits en el arreglo de origen contra los bits en el arreglo de referencia, registra el número de bit de cada desigualdad en el arreglo de resultados y cambia el valor de referencia para que coincida con el valor del bit de origen correspondiente.

La instrucción DDT opera en la memoria de datos contiguos. Vea la página, *Cómo ver un arreglo como un bloque de memoria*, para obtener más información B-4.

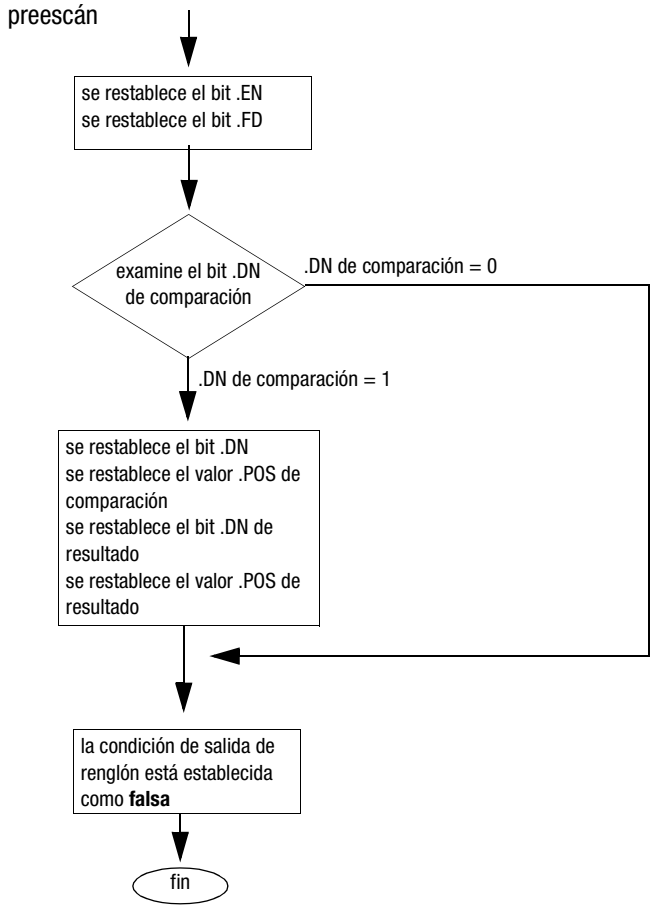
La diferencia entre las instrucciones DDT y FBC es que cada vez que la instrucción DDT encuentra una desigualdad, la instrucción DDT cambia el bit de referencia para que coincida con el bit de origen. La instrucción FBC no cambia el bit de referencia.

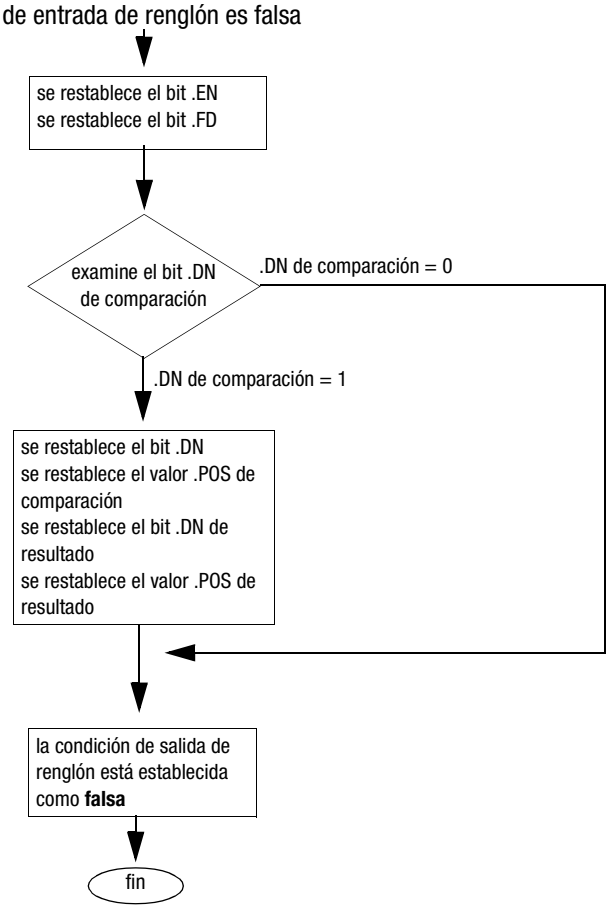
Cómo seleccionar el modo buscar

Si usted desea detectar:	Seleccione este modo:
Una desigualdad a la vez	Establezca el bit .IN en la estructura CONTROL de comparación. Cada vez que la condición de entrada de renglón va de falso a verdadero, la instrucción DDT busca la próxima desigualdad entre los arreglos de origen y referencia. Cuando la instrucción encuentra una desigualdad, ésta establece el bit .FD, registra la posición de la desigualdad y detiene la ejecución.
Todas las desigualdades	Restablezca el bit .IN en la estructura CONTROL de comparación. Cada vez que la condición de entrada de renglón va de falso a verdadero, la instrucción DDT busca todas las desigualdades entre los arreglos de origen y referencia.

Ejecución:

Condición:	Acción:

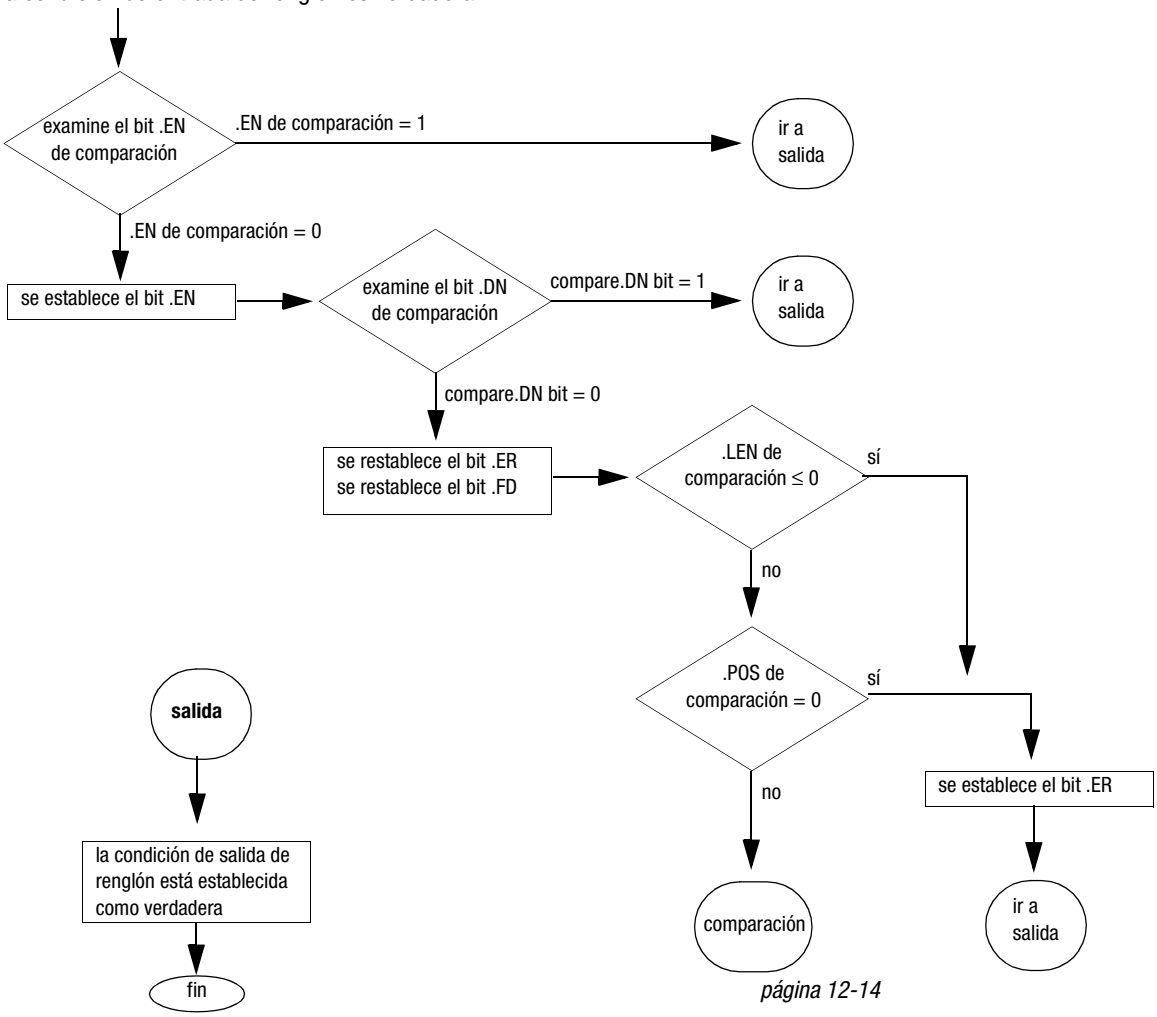

preescán
se restablece el bit .EN
se restablece el bit .FD
examine el bit .DN de comparación
.DN de comparación = 0
.DN de comparación = 1
se restablece el bit .DN
se restablece el valor .POS de comparación
se restablece el bit .DN de resultado
se restablece el valor .POS de resultado
la condición de salida de renglón está establecida como **falsa**
fin

Condición:	**Acción:**
la condición de entrada de renglón es falsa	

se restablece el bit .EN
se restablece el bit .FD
examine el bit .DN de comparación
.DN de comparación = 0
.DN de comparación = 1
se restablece el bit .DN
se restablece el valor .POS de comparación
se restablece el bit .DN de resultado
se restablece el valor .POS de resultado
la condición de salida de renglón está establecida como **falsa**
fin

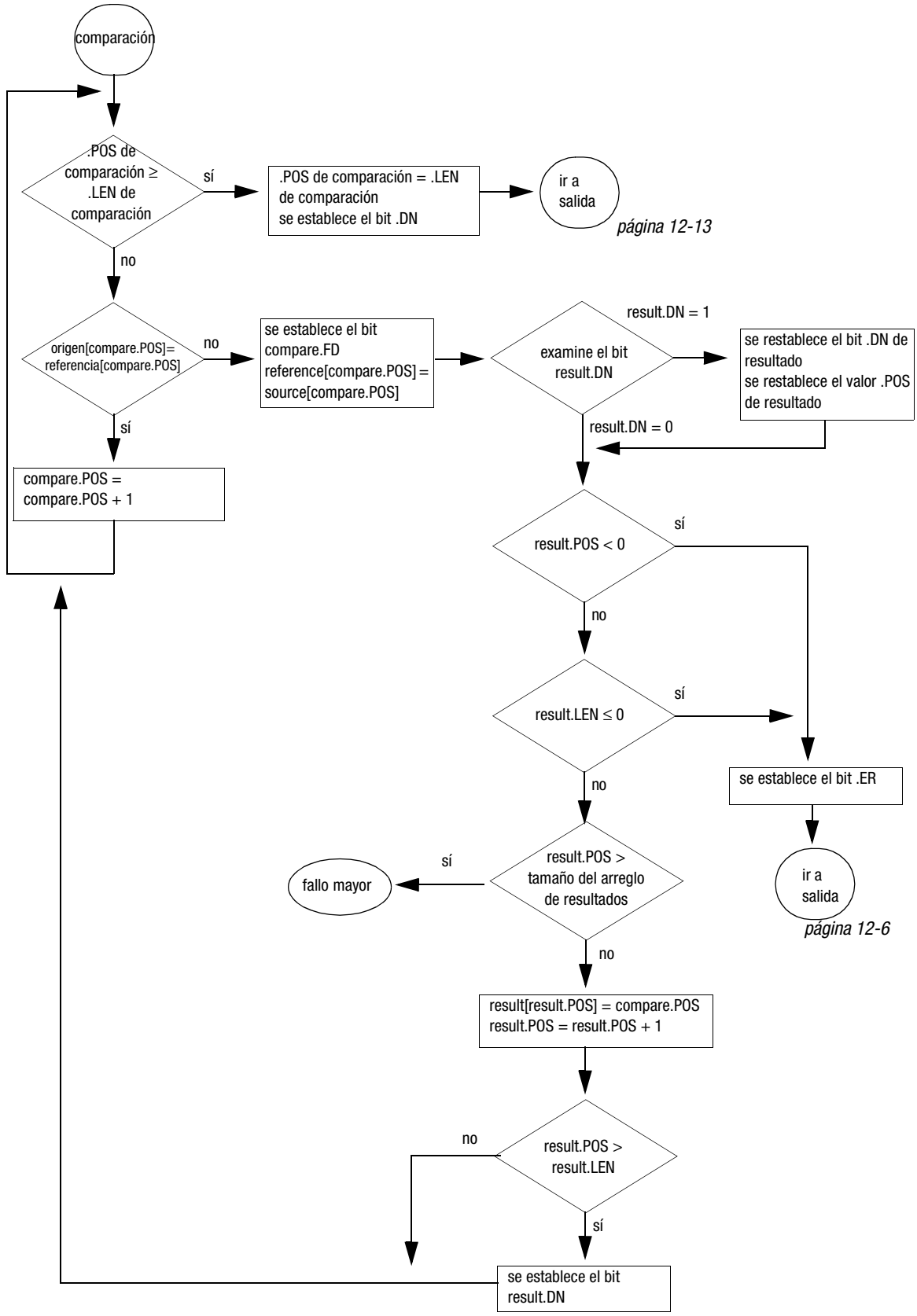
Condición: **Acción:**

la condición de entrada de renglón es verdadera



página 12-14

Condición: **Acción:**

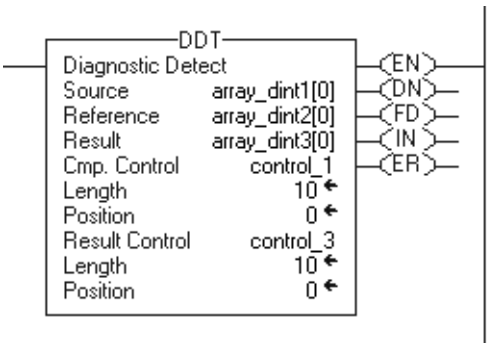


Indicadores de estado aritmético: no afectados

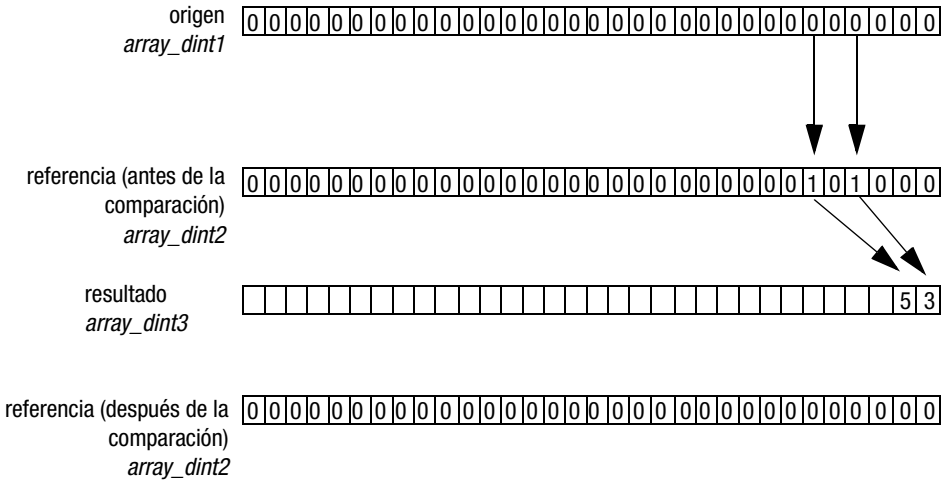
Condiciones de fallo:

Ocurrirá un fallo mayor si:	Tipo de fallo:	Código de fallo:
Result.POS > tamaño del arreglo de resultados	4	20

Ejemplo de DDT:



Una vez habilitada, la instrucción DDT compara el origen *array_dint1* contra la referencia *array_dint2* y almacena la ubicaciones de las desigualdades en el resultado *array_dint3*. El controlador también cambia los bits desiguales en la referencia *array_dint2* para que coincidan con el origen *array_dint1*.



Otros formatos:

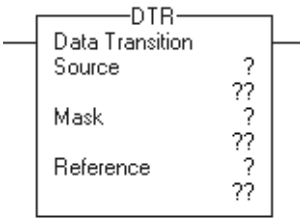
Formato:	Sintaxis:
texto neutro	`DDT(source,reference,result,cmp_control,length,position,result_control,length,position);`
texto ASCII	`DDT source reference result cmp_control length position result_control length position`

Instrucciones relacionadas: FBC, DTR

Data Transitional (DTR)

La instrucción DTR es una instrucción de entrada.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	DINT	valor inmediato tag	arreglo que se compara contra la referencia
Máscara	DINT	valor inmediato tag	qué bits se bloquean o se pasan
Referencia	DINT	tag	arreglo que se compara contra el origen

Descripción: La instrucción DTR pasa el valor de origen mediante una máscara y compara el resultado con el valor de referencia. La instrucción DTR también escribe el valor de origen enmascarado en el valor de referencia para la próxima comparación. El origen no se cambia.

Un número “1” en la máscara significa que se pasa el bit de datos. Un número “0” en la máscara significa que se bloquea el bit de datos.

Cuando el origen enmascarado difiere de la referencia, la condición de salida de renglón se hace verdadera durante un escán. Cuando el origen enmascarado es idéntico al de la referencia, la condición de salida de renglón es falsa.

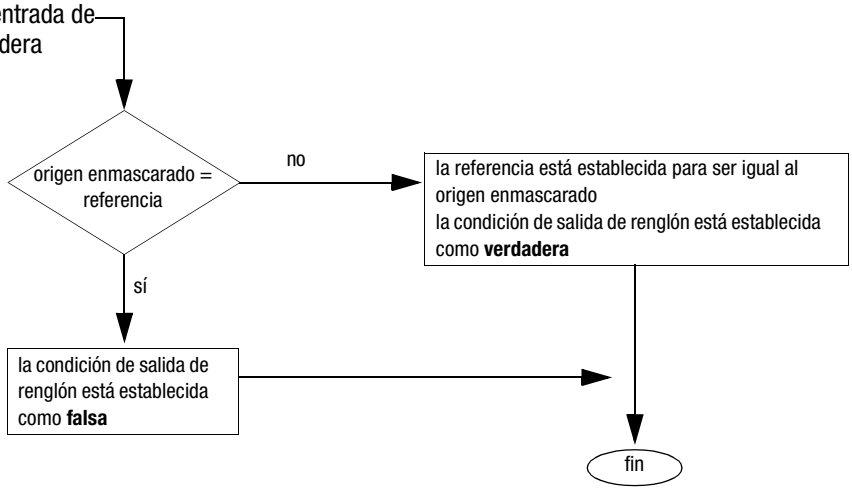
ATENCION: La programación en línea con esta instrucción puede ser peligrosa. Si el valor de referencia es diferente que el valor de origen, la condición de salida de renglón es verdadera. Tenga cuidado si inserta esta instrucción cuando el procesador está en el modo marcha o marcha remota.

Cómo introducir un valor de máscara inmediato

Cuando introduce una máscara, el software de programación vuelve a establecer los valores decimales predeterminados. Si desea introducir una máscara usando otro formato, ponga el prefijo correcto antes del valor.

Prefijo:	Descripción:
16#	hexadecimal por ejemplo; 16#0F0F
8#	octal por ejemplo; 8#16
2#	binario por ejemplo; 2#00110011

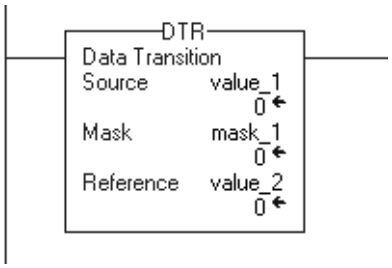
Ejecución:

Condición:	Acción:
preescán	La referencia = máscara AND de origen. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La referencia = máscara AND de origen. La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	

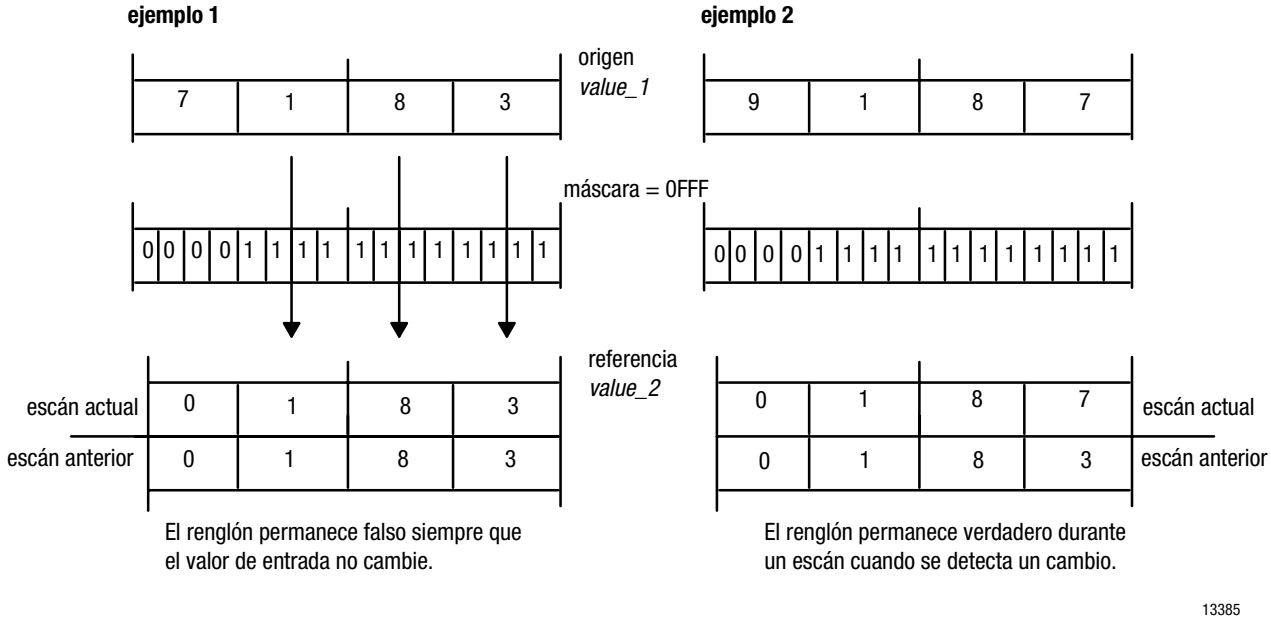
Indicadores de estado aritmético: no afectados

Condiciones de fallo: ninguna

Ejemplo de DTR:



Una vez habilitada, la instrucción DTR enmascara *value_1*. Si hay una diferencia entre los dos valores, la condición de salida de renglón se establece como verdadera.


ejemplo 1
ejemplo 2
origen
value_1
7 1 8 3
9 1 8 7
máscara = 0FFF
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
referencia
value_2
escán actual
escán anterior
0 1 8 3
0 1 8 3
0 1 8 7
0 1 8 3
escán actual
escán anterior
El renglón permanece falso siempre que el valor de entrada no cambie.
El renglón permanece verdadero durante un escán cuando se detecta un cambio.

13385

Una cifra 0 en la máscara no cambia el bit.

Otros formatos:

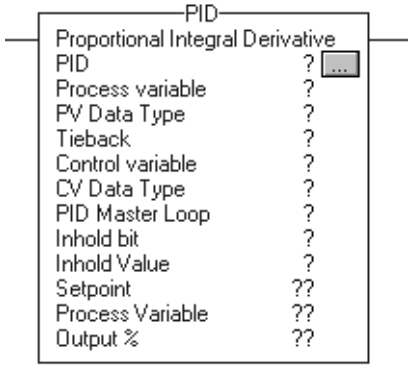
Formato:	Sintaxis:
texto neutro	`DTR(source,mask,reference);`
texto ASCII	`DTR source mask reference`

Instrucciones relacionadas: FBC, DDT

Proportional Integral Derivative (PID)

La instrucción PID es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
PID	PID	estructura	Estructura PID
Variable de proceso	SINT INT DINT REAL	tag	valor que desea controlar
Valor retenido	SINT INT DINT REAL	valor inmediato tag	*(opcional)* salida de una estación manual/automática de hardware que evita la salida del controlador introduzca 0 si no desea este parámetro
Variable de control	SINT INT DINT REAL	tag	valor que va al dispositivo de control final (válvula, amortiguador, etc.) Si usted usa la banda muerta, la variable de control debe ser REAL, si no se forzará a 0 cuando el error se encuentre dentro de la banda muerta
Lazo maestro PID	PID	estructura	opcional tag PID para el PID maestro Si realiza el control en cascada y este PID es un lazo de esclavo, introduzca el nombre del PID maestro introduzca 0 si no desea este parámetro
Bit mantenido	BOOL	tag	opcional estado actual del bit mantenido de un canal de salida analógico 1756 para aceptar el reinicio sin perturbaciones introduzca 0 si no desea este parámetro
Valor mantenido	SINT INT DINT REAL	tag	opcional valor de lectura de datos de un canal de salida analógico 1756 para aceptar el reinicio sin perturbaciones introduzca 0 si no desea este parámetro
Punto de ajuste			para visualización solamente valor actual del punto de ajuste
Variable de proceso			para visualización solamente valor actual de la variable de proceso escalada
Porcentaje de salida			para visualización solamente valor de porcentaje de salida actual

Estructura PID Especifique una estructura PID única para cada instrucción PID.

Mnemónico:	Tipo de datos:	Descripción:	
.CTL	DINT	El miembro .CTL proporciona acceso a los miembros de estado (bits) en una palabra de 32 bits. La instrucción PID establece los bits 07 a 15.	
		Este bit:	**Es este miembro:**
		31	.EN
		30	.CT
		29	.CL
		28	.PVT
		27	.DOE
		26	.SWM
		25	.CA
		24	.MO
		23	.PE
		22	.NDF
		21	.NOBC
		20	.NOZC
		Este bit:	**Es este miembro que la instrucción PID establece:**
		15	.INI
		14	.SPOR
		13	.OLL
		12	.OLH
		11	.EWD
		10	.DVNA
		09	.DVPA
		08	.PVLA
		07	.PVHA
.SP	REAL	punto de ajuste	
.KP	REAL	independiente	ganancia proporcional (sin unidad)
		dependiente	ganancia del controlador (sin unidad)
.KI	REAL	independiente	ganancia integral (1/seg)
		dependiente	tiempo de restablecimiento (minutos por repetición)
.KD	REAL	independiente	ganancia de derivada (segundos)
		dependiente	tiempo de régimen (minutos)
.BIAS	REAL	porcentaje de prealimentación o polarización	
.MAXS	REAL	valor de escalado máximo de unidad de ingeniería	
.MINS	REAL	valor de escalado mínimo de unidad de ingeniería	
.DB	REAL	unidades de ingeniería de banda muerta	

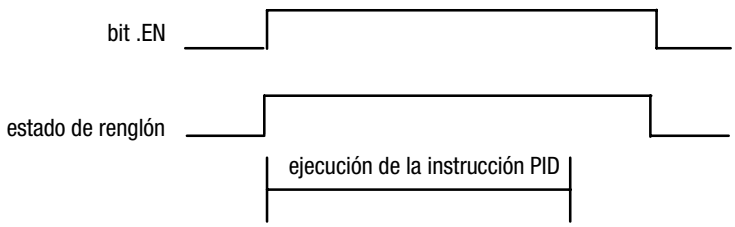
Mnemónico:	Tipo de datos:	Descripción:
.SO	REAL	porcentaje de salida establecido
.MAXO	REAL	límite máximo de salida (% de salida)
.MINO	REAL	límite mínimo de salida (% de salida)
.UPD	REAL	tiempo de actualización del lazo (segundos)
.PV	REAL	valor PV escalado
.ERR	REAL	valor de error escalado
.OUT	REAL	porcentaje de salida
.PVH	REAL	límite de alarma alta de la variable de proceso
.PVL	REAL	límite de alarma baja de la variable de proceso
.DVP	REAL	límite de alarma de desviación positiva
.DVN	REAL	límite de alarma de desviación negativa
.PVDB	REAL	banda muerta de la alarma de variable de proceso
.DVDB	REAL	banda muerta de la alarma de desviación
.MAXI	REAL	valor PV máximo (entrada sin escala)
.MINI	REAL	valor PV mínimo (entrada sin escala)
.TIE	REAL	valor retenido para el control manual
.MAXCV	REAL	valor CV máximo (que corresponde al 100 %)
.MINCV	REAL	valor CV mínimo (que corresponde al 0 %)
.MINTIE	REAL	valor retenido mínimo (que corresponde al 100 %)
.MAXTIE	REAL	valor retenido máximo (que corresponde al 0 %)

Mnemónico:	Tipo de datos:	Descripción:	
.DATA	REAL[17]	El miembro .DATA almacena:	
		Elemento:	**Descripción:**
		.DATA[0]	acumulación integral
		.DATA[1]	valor temporal de uniformidad derivada
		.DATA[2]	valor .PV anterior
		.DATA[3]	valor .ERR anterior
		.DATA[4]	valor .SP válido anterior
		.DATA[5]	constante de escala de porcentaje
		.DATA[6]	constante de escala .PV
		.DATA[7]	constante de escala derivada
		.DATA[8]	valor .KP anterior
		.DATA[9]	valor .KI anterior
		.DATA[10]	valor .KD anterior
		.DATA[11]	ganancia independiente .KP
		.DATA[12]	ganancia independiente .KI
		.DATA[13]	ganancia independiente .KD
		.DATA[14]	valor .CV anterior
		.DATA[15]	constante sin escala .CV
		.DATA[16]	constante sin escala de valor retenido
.EN	BOOL	habilitado	
.CT	BOOL	tipo en cascada (0 = esclavo; 1 = maestro)	
.CL	BOOL	lazo en cascada (0 = no; 1 = sí)	
.PVT	BOOL	seguimiento de variable de proceso (0 = no; 1 = sí)	
.DOE	BOOL	derivada de (0 = PV; 1 = error)	
.SWM	BOOL	modo manual de software (0 = no-auto; 1 = sí-interr. manual)	
.CA	BOOL	acción de control (0 significa E = SP-PV; 1 significa E = PV-SP)	
.MO	BOOL	modo estación (0 = automático; 1 = manual)	
.PE	BOOL	ecuación PID (0 = independiente; 1 = dependiente)	
.NDF	BOOL	uniformidad derivada (0 = no; 1 = sí)	
.NOBC	BOOL	cálculo sin datos originales de polarización (0 = no; 1 = sí)	
.NOZC	BOOL	paso por cero para banda muerta (0 = no; 1 = sí para banda muerta)	
.INI	BOOL	PID inicializado (0 = no; 1 = sí)	
.SPOR	BOOL	punto de ajuste fuera de rango (0 = no; 1 = sí)	
.OLL	BOOL	CV se encuentra por debajo del límite mínimo de salida (0 = no; 1 = sí)	
.OLH	BOOL	CV se encuentra por encima del límite máximo de salida (0 = no; 1 = sí)	
.EWD	BOOL	error se encuentra dentro de la banda muerta (0 = no; 1 = sí)	
.DVNA	BOOL	alarma baja de desviación (0 = no; 1 = sí)	

Mnemónico:	Tipo de datos:	Descripción:
.DVPA	BOOL	alarma alta de desviación (0 = no; 1 = sí)
.PVLA	BOOL	alarma baja de PV (0 = no; 1 = sí)
.PVHA	BOOL	alarma alta de PV (0 = no; 1 = sí)

Descripción: La instrucción PID controla una variable de proceso tal como el flujo, la presión, la temperatura o el nivel. La instrucción PID típicamente recibe la variable de proceso (PV) de un módulo de entrada analógica y modula una salida de variable de control (CV) en un módulo de salida analógica para mantener la variable de proceso en el punto de ajuste deseado.

El bit .EN indica el estado de ejecución. Se establece el bit .EN cuando la condición de entrada de renglón cambia de falso a verdadero. Se restablece el bit .EN cuando la condición de entrada de renglón se hace falsa. La instrucción PID no usa un bit .DN. La instrucción PID ejecuta cada escán siempre que la condición de entrada de renglón sea verdadera.



Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: no afectados

Condiciones de fallo:

Ocurrirá un fallo menor si:	Tipo de fallo:	Código de fallo:
.UPD ≤ 0	4	35
punto de ajuste fuera de rango	4	36

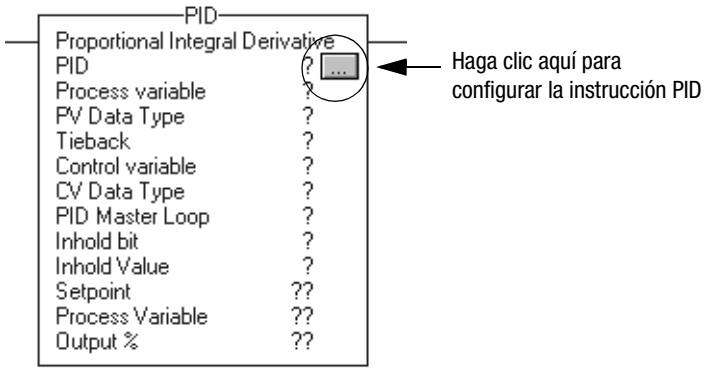
Importante: Estos fallos eran fallos mayores en el procesador PLC-5.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`PID(pv,pv_type,tieback,cv,cv_type,master,inhold_bit,inhold_value);`
texto ASCII	`PID pv pv_type tieback cv cv_type master inhold_bit inhold_value`

Cómo configurar una instrucción PID

Después de introducir la instrucción PID y especificar la estructura PID, usted usa las fichas de configuración para especificar cómo debe funcionar la instrucción PID.



Cómo especificar el ajuste

Seleccione la ficha Ajuste. Los cambios se realizan después de hacer clic en otro campo, hace clic en OK, hacer clic en Aplicar o hacer clic en Enter.

En este campo	Especifique:
Punto de ajuste (SP)	Introduzca un valor de punto de ajuste (.SP).
Porcentaje de salida establecido	Introduzca un porcentaje de salida (.SO). En el modo manual de software, este valor se usa para la salida. En el modo automático, este valor muestra el porcentaje de salida.
Polarización de salida	Introduzca un porcentaje de polarización de salida (.BIAS).
Ganancia proporcional (K_p)	Introduzca la ganancia proporcional (.KP). Para las ganancias independientes, ésta es la ganancia proporcional (sin unidad). Para las ganancias dependientes, ésta es la ganancia del controlador (sin unidad).
Ganancia integral (K_i)	Introduzca la ganancia integral (.KI). Para las ganancias independientes, ésta es la ganancia integral (1/seg). Para las ganancias dependientes, este valor es el tiempo de restablecimiento (minutos por repetición).
Tiempo de derivada (K_d)	Introduzca la ganancia de derivada (.KD). Para las ganancias independientes, ésta es la ganancia de derivada (segundos). Para las ganancias dependientes, este valor es el tiempo de régimen (minutos).
Modo manual	Seleccione manual (.MO) o manual mediante software (.SWM). El modo manual anula el modo manual mediante software si se seleccionan los dos.

Cómo especificar la configuración

Seleccione la ficha Configuración. Usted debe hacer clic en OK o Aplicar para que los cambios se realicen.

En este campo	Especifique:
Ecuación PID	Seleccione ganancias independientes o ganancias dependientes (.PE). Use las ganancias independientes cuando desee que tres ganancias (P, I y D) realicen la operación independientemente. Use las ganancias dependientes cuando desee una ganancia de controlador general que afecte los tres términos (P, I y D).
Acción de control	Seleccione E = PV-SP o E = SP-PV para la acción de control (.CA).
Derivada de	Seleccione PV o error (.DOE). Use la derivada de PV para eliminar los picos de salida que resultan de los cambios del punto de ajuste. Use la derivada de error para obtener respuestas rápidas de cambios del punto de ajuste cuando al algoritmo puede tolerar los sobreimpulsos.
Tiempo de actualización del lazo	Introduzca el tiempo de actualización (.UPD) para la instrucción (mayor que o igual a 0.01 segundos).
Límite alto CV	Introduzca un límite alto para la variable de control (.MAXO).
Límite bajo CV	Introduzca un límite bajo para la variable de control (.MINO).
Valor de la banda muerta	Introduzca un valor de banda muerta (.DB).
Sin uniformidad de derivada	Habilite o inhabilite esta selección (.NDF).
Sin cálculo de polarización	Habilite o inhabilite esta selección (.NOBC).
Sin paso por cero en la banda muerta	Habilite o inhabilite esta selección (.NOZC).
Seguimiento PV	Habilite o inhabilite esta selección (.PVT).
Lazo en cascada	Habilite o inhabilite esta selección (.CL).
Tipo cascada	Si se habilita un lazo en cascada, seleccione un esclavo o maestro (.CT).

Cómo especificar alarmas

Seleccione la ficha Alarmas. Usted debe hacer clic en OK o Aplicar para que los cambios se realicen.

En este campo	Especifique:
PV alto	Introduzca un valor de alarma alta PV (.PVH).
PV bajo	Introduzca un valor de alarma baja PV (.PVL).
Banda muerta PV	Introduzca un valor de banda muerta de alarma PV (.PVDB).
desviación positiva	Introduzca un valor de desviación positivo (.DVP).
desviación negativa	Introduzca un valor de desviación negativo (.DVN).
banda muerta de desviación	Introduzca un valor de banda muerta de alarma de desviación (.DVDB).

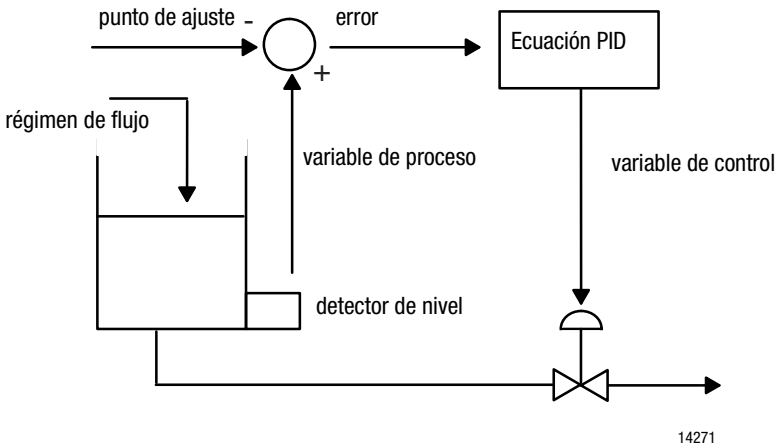
Cómo especificar la escala

Seleccione la ficha Escala. Usted debe hacer clic en OK o Aplicar para que los cambios se realicen.

En este campo	Especifique:
PV máximo sin escala	Introduzca un valor PV máximo (.MAXI) que sea igual al valor máximo sin escala recibido del canal de entrada analógico para el valor PV.
PV mínimo sin escala	Introduzca un valor PV mínimo (.MINI) que sea igual al valor mínimo sin escala recibido del canal de entrada analógico para el valor PV.
Máximo de unidades de ingeniería PV	Introduzca el máximo de unidades de ingeniería correspondientes a .MAXI (.MAXS).
Mínimo de unidades de ingeniería PV	Introduzca el mínimo de unidades de ingeniería correspondientes a .MINI (.MINS).
CV máximo	Introduzca un valor CV máximo correspondiente al 100 % (.MAXCV).
CV mínimo	Introduzca un valor CV mínimo correspondiente al 0 % (.MINCV).
Valor retenido máximo	Introduzca un valor retenido máximo (.MAXTIE) que sea igual al valor máximo sin escala recibido del canal de entrada analógica para el valor retenido.
Valor retenido mínimo	Introduzca un valor retenido mínimo (.MINTIE) que sea igual al valor mínimo sin escala recibido del canal de entrada analógica para el valor retenido.
PID inicializado	Si cambia las constantes de escala durante el modo marcha, desactive este campo para reinicializar los valores sin escala internos (.INI).

Cómo usar las instrucciones PID

El control de lazo cerrado PID mantiene una variable de proceso en un punto de ajuste deseado. La figura siguiente muestra un ejemplo de régimen de flujo/nivel de fluido:



En el ejemplo anterior, el nivel en el tanque se compara con el punto de ajuste. Si el nivel sobrepasa el punto de ajuste, la ecuación PID incrementa la variable de control que causa la abertura de la válvula de salida del tanque, reduciendo así el nivel en el tanque.

La ecuación PID usada en la instrucción PID es una ecuación de formato posicional que ofrece la opción de usar ganancias independientes o ganancias dependientes. Cuando se usan las ganancias independientes, las ganancias proporcionales, integrales y de derivada afectan solamente sus términos proporcionales, integrales o de derivada respectivamente. Cuando se usan las ganancias dependientes, la ganancia proporcional se reemplaza con una ganancia de controlador que afecta los tres términos. Usted puede usar cualquiera de las ecuaciones para realizar el mismo tipo de control. Se proporcionan dos tipos de ecuación para permitirle usar el tipo con que está más familiarizado.

Opción de ganancias:	**Derivada de:**	**Ecuación:**
Ganancias dependientes (norma ISA)	error (E)	$$CV = K_C\left[E + \frac{1}{T_i}\int_0^t Edt + T_d\frac{dE}{dt}\right] + BIAS$$
	variable de proceso (PV)	E = SP – PV $$CV = K_C\left[E + \frac{1}{T_i}\int_0^t Edt - T_d\frac{dPV}{dt}\right] + BIAS$$ E = PV – SP $$CV = K_C\left[E + \frac{1}{T_i}\int_0^t Edt + T_d\frac{dPV}{dt}\right] + BIAS$$
ganancias independientes	error (E)	$$CV = K_PE + K_i\int_0^t Edt + K_d\frac{dE}{dt} + BIAS$$
	variable de proceso (PV)	E = SP – PV $$CV = K_PE + K_i\int_0^t Edt - K_d\frac{dPV}{dt} + BIAS$$ E = PV – SP $$CV = K_PE + K_i\int_0^t Edt + K_d\frac{dPV}{dt} + BIAS$$

donde:

Variable:	Descripción:
K_P	ganancia proporcional (sin unidad) $K_p = K_c$ sin unidad
K_i	ganancia integral (segundos $^{-1}$) Para convertir entre K_i (ganancia integral) y T_i (tiempo de restablecimiento), use: $K_i = \frac{K_C}{60T_i}$
K_d	ganancia de derivada (segundos) Para convertir entre K_d (ganancia de derivada) y T_d (tiempo de régimen), use: $K_d = K_c (T_d) 60$
K_C	ganancia del controlador (sin unidad)
T_i	tiempo de restablecimiento (minutos/repetición)
T_d	tiempo de régimen (minutos)
SP	punto de ajuste
PV	variable de proceso
E	error [(SP – PV) o (PV – SP)]
BIAS	prealimentación o polarización
CV	variable de control
dt	tiempo de actualización del lazo

Si usted no desea usar un término determinado de la ecuación PID, ponga la ganancia en cero. Por ejemplo, si no desea una acción de derivada, ponga K_d o T_d en cero.

Bloqueo de acción integral y transferencia sin perturbaciones de manual a automático

La instrucción PID evita automáticamente la acción integral impidiendo que el término integral se acumule cuando la salida CV llegue a sus valores máximo o mínimo según los establecen .MAXO y .MIN. El término integral acumulado permanece fijo hasta que la salida CV cae por debajo del límite máximo o sube por encima del límite mínimo. La acumulación integral normal se reanuda automáticamente.

La instrucción PID es compatible con dos modos manuales de control:

Modo manual de control:	Descripción:
manual mediante software (.SWM)	también conocido como modo de salida establecida permite que el usuario establezca el porcentaje de salida mediante el software El valor de salida establecida (.SO) se usa como la salida del lazo. El valor de salida establecida típicamente proviene de una entrada de operador desde un dispositivo de interface de operador.
manual (.MO)	halla el valor retenido, como entrada, y ajusta sus variables internas para generar el mismo valor en la salida. La entrada de valor retenido a la instrucción PID se escala al 0 – 100 % según los valores de .MINTIE y .MAXTIE y se usa como la salida del lazo. La entrada del valor retenido típicamente proviene de la salida de una estación manual/automática de hardware, la cual evita la salida del controlador. **Nota:** El modo manual anula el modo manual mediante software si ambos bits de modo están establecidos.

La instrucción PID también proporciona automáticamente las transferencias sin perturbaciones desde el modo manual mediante software hacia el modo automático o desde el modo manual al modo automático. La instrucción PID calcula en base a datos originales el valor del término de acumulación integral requerido para hacer que la salida CV siga el valor de salida establecida (.SO) en el modo manual mediante software o la entrada del valor retenido en el modo manual. De esta manera, cuando el lazo cambia al modo automático, la salida CV se inicia a partir de la salida establecida o el valor retenido y no ocurre una perturbación en el valor de salida.

La instrucción PID también puede proporcionar automáticamente una transferencia sin perturbaciones del modo manual al modo automático aun cuando no se usa el control integral (por ejemplo, $K_i = 0$). En tal caso, la instrucción modifica el término .BIAS para hacer que la salida CV siga la salida establecida o los valores retenidos. Cuando se reanuda el control automático, el término .BIAS mantiene su último valor. Puede inhabilitar el cálculo en base a datos originales del término .BIAS estableciendo el bit .NOBC en la estructura de datos PID. Tome nota de que si establece el bit .NOBC como verdadero, la instrucción PID ya no proporciona la transferencia sin perturbaciones desde el modo manual al modo automático cuando no se usa el control integral.

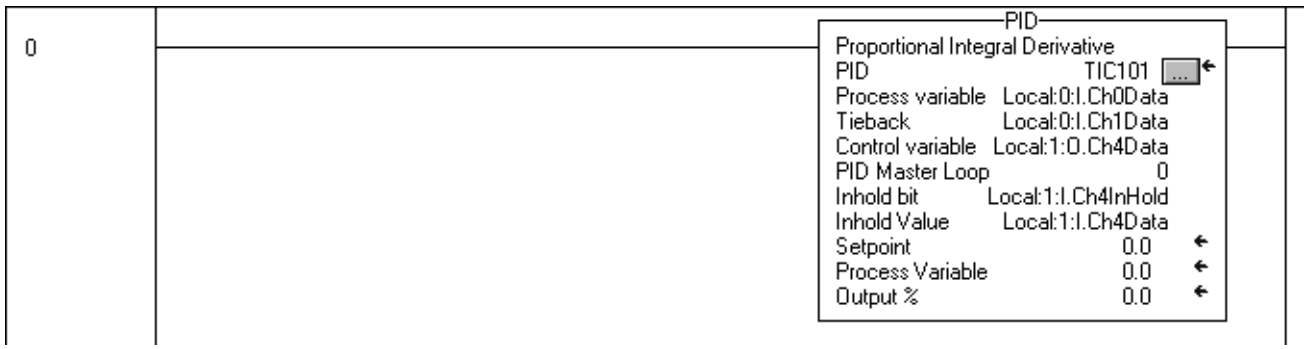
Temporización de la instrucción PID

La instrucción PID y el muestreo de la variable de proceso deben actualizarse según un régimen periódico. Este tiempo de actualización está relacionado al proceso físico que se controla. Para los lazos muy lentos, tales como los lazos de temperatura, un tiempo de actualización de una vez por segundo o más largo es generalmente suficiente para obtener un control adecuado. Los lazos más rápidos, tales como los lazos de presión o flujo, pueden requerir un tiempo de actualización tal como una vez cada 250 milisegundos. Hay casos raros, como el control de tensión de una bobina desen-

rolladora, que requieren actualizaciones de datos a una velocidad de una vez cada 10 milisegundos o más rápido.

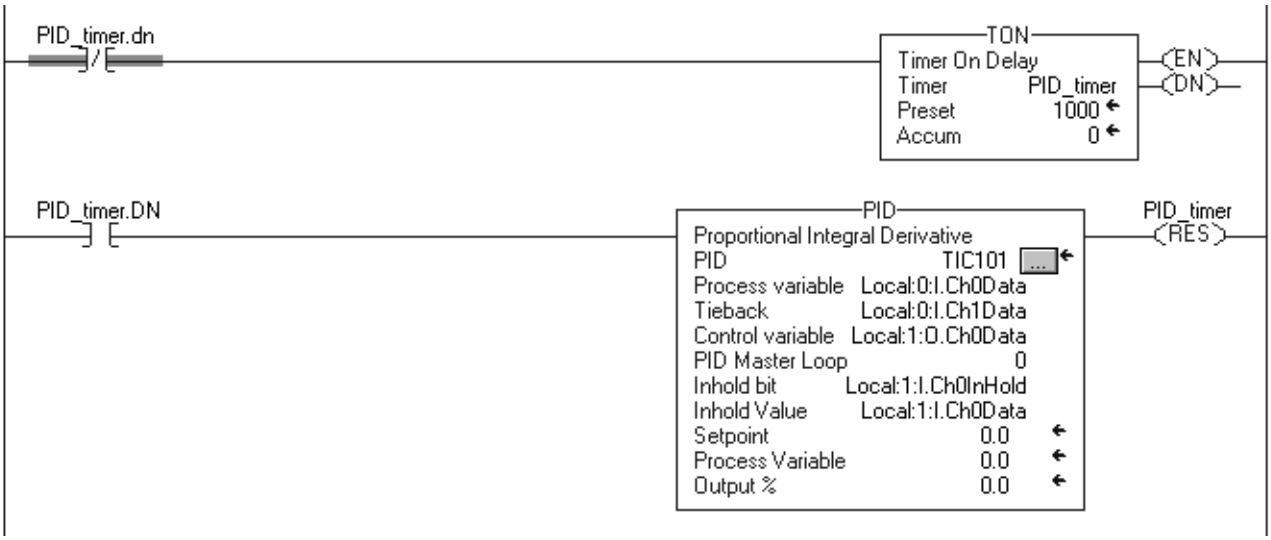
Puesto que la instrucción PID usa una base de tiempo para calcular, usted debe sincronizar la ejecución de esta instrucción con el muestreo de la variable de proceso (PV).

La manera más fácil de ejecutar la instrucción PID es colocar ésta en una tarea periódica. Establezca el tiempo de actualización del lazo (.UPD) para que sea igual al régimen de la tarea periódica y asegúrese de que la instrucción PID se ejecuta durante cada escán de la tarea periódica. Por ejemplo, use un renglón de lógica de escalera sin condición.



Al usar una tarea periódica, asegúrese de que la entrada analógica usada para la variable de proceso se actualiza y envía al procesador a un régimen sumamente más rápido que el régimen de la tarea periódica. En el caso más idóneo, la variable de proceso se debe enviar al procesador por lo menos cinco a diez veces más rápido que el régimen de la tarea periódica. Esto minimiza la diferencia de tiempo entre los muestreos reales de la variable de proceso y la ejecución del lazo PID. Por ejemplo, si el lazo PID se encuentra en una tarea periódica de 250 milisegundos, use un tiempo de actualización del lazo de 250 milisegundos (.UPD = .25) y configure el módulo de entrada analógica para que produzca datos por lo menos cada 25 a 50 msegs.

Otro método menos preciso de ejecutar una instrucción PID es colocar la instrucción en una tarea continua y usar un bit de efectuado del temporizador para activar la ejecución de la instrucción PID.

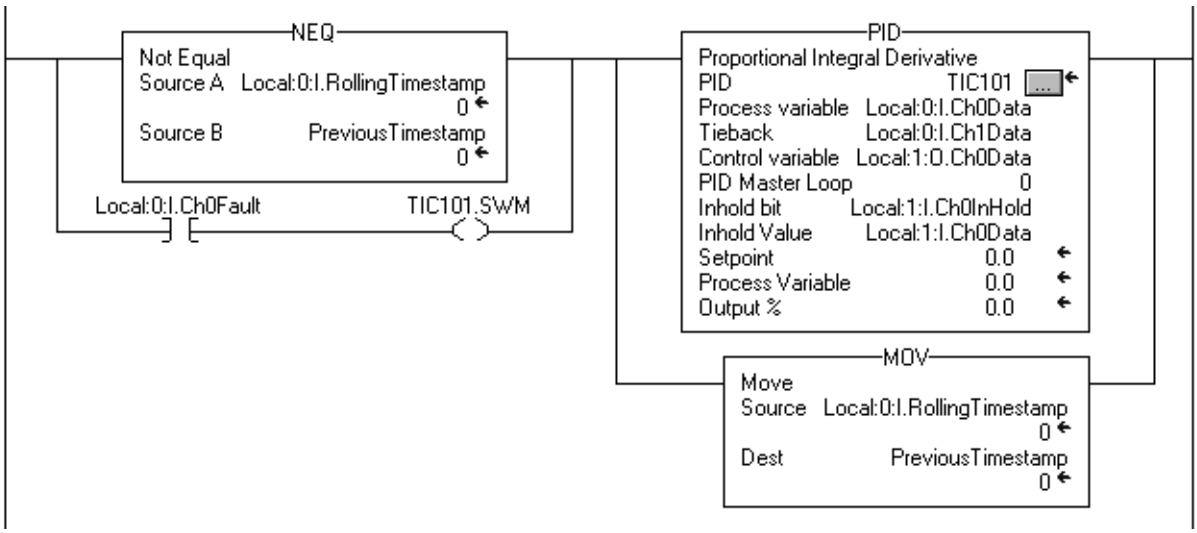


Con este método, el tiempo de actualización del lazo de la instrucción PID se debe establecer para que sea igual al valor predeterminado del temporizador. Igual al uso de una tarea periódica, se debe establecer el módulo de entrada analógica para producir la variable de proceso a un régimen sumamente más rápido que el tiempo de actualización del lazo. Debe usar el método de temporizador de ejecución PID solamente para los lazos con tiempos de actualización del lazo que sean por lo menos varias veces más largos que el tiempo de ejecución en el peor de los casos para la tarea continua.

La manera más precisa de ejecutar una instrucción PID es usar la función de muestreo en tiempo real (RTS) de los módulos de entrada analógica 1756. El módulo de entrada analógica realiza un muestreo de sus entradas al régimen de muestreo en tiempo real que se configuró al instalar el módulo. Cuando expira el período de muestreo en tiempo real del módulo, éste actualiza sus entradas y actualiza un sello de hora continuo (representado por el miembro RollingTimestamp de la estructura de datos de entrada analógica) producido por el módulo. El sello de hora tiene un rango de 0 – 32.767 milisegundos. Monitoree el sello de hora. Cuando cambia, se ha recibido un nuevo muestreo de la variable de proceso. Cada vez que cambie un sello de hora, ejecute la instrucción PID una vez. Puesto que el muestreo de la variable de proceso es accionado por el módulo de entrada analógica, el tiempo de muestreo de entrada es muy preciso y el tiempo de actualización del lazo usado por la instrucción PID debe establecerse para que sea igual al tiempo RTS del módulo de entrada analógica.

Para asegurarse de no pasar por alto los muestreos de la variable de proceso, ejecute la lógica a un régimen más rápido que el tiempo RTS. Por ejemplo, si el tiempo RTS es 250 msegs, puede colocar la lógica PID en una tarea periódica que se ejecuta cada 100 msegs asegurando así que nunca se pasa por alto un muestreo. Hasta es posible colocar la lógica PID en una tarea continua siempre que se asegure de que la lógica se actualiza más frecuentemente que una vez cada 250 milisegundos.

Una ejemplo del método RTS de ejecución se muestra a continuación. La ejecución de la instrucción PID depende de la recepción de nuevos datos de entrada analógica. Si el módulo de entrada anaIógica entra en fallo o se retira, el controlador ya no recibe los sellos de hora continuos y el lazo PID detiene la ejecución. Debe monitorear el bit de estado de la entrada analógica PV y, si presenta un estado deficiente, fuerce el lazo en el modo manual mediante software y ejecute el lazo durante cada escán. Esto permite que el operador pueda cambiar manualmente la salida del lazo PID.



Reinicio sin perturbaciones

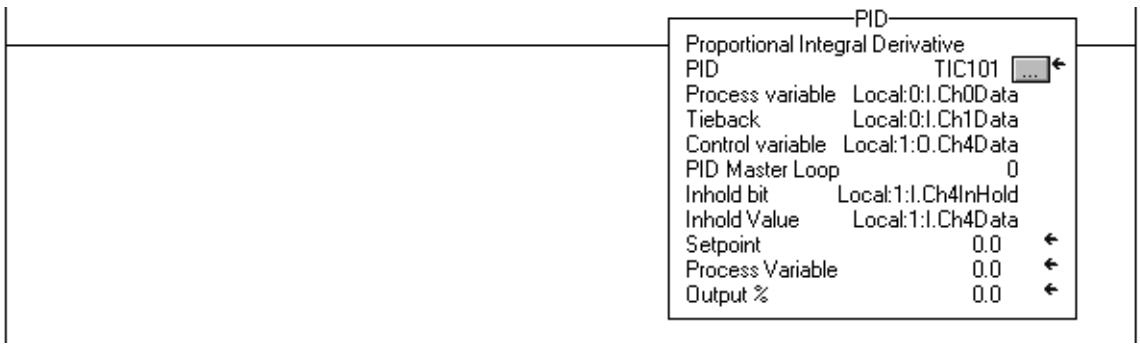
La instrucción PID puede interactuar con los módulos de salida analógica 1756 para ser compatible con un reinicio sin perturbaciones cuando el controlador cambia del modo de programación al modo marcha o cuando se enciende el controlador.

Cuando un módulo de salida analógica 1756 pierde comunicación con el controlador o cuando detecta que el controlador está en el modo de programación, el módulo de salida analógica establece sus salidas en los valores de condición de fallo que usted especificó al configurar el módulo. Cuando el controlador retorna al modo marcha o cuando restablece la comunicación con el módulo de salida analógica, puede hacer que la instrucción PID restablezca automáticamente su salida de variable de control para ser igual a la salida analógica usando los parámetros del bit mantenido y valor mantenido en la instrucción PID.

Para establecer un reinicio sin perturbaciones:

Haga lo siguiente:	**Detalles:**
Configurar el canal del módulo de salida analógica 1756 que recibe la variable de control desde la instrucción PID	Selecciona la casilla de verificación “retener hasta inicialización” en la página de propiedades para el canal específico del módulo. Esto indica al módulo de salida analógica que, cuando el controlador retorna al modo marcha o restablece la comunicación con el módulo, el módulo debe mantener la salida analógica en su valor actual hasta que el valor enviado desde el controlador coincida (dentro del 0,1 % del intervalo) con el valor actual usado por el canal de salida. La salida del controlador se aumentará al valor de salida actualmente mantenido usando el término .BIAS. Este aumento es similar a la transferencia sin perturbaciones automática.
Introducir el tag del bit mantenido y el tag del valor mantenido en la instrucción PID	El módulo de salida analógica 1756 retorna dos valores para cada canal en su estructura de datos de entrada. El bit de estado InHold (por ejemplo, .Ch2InHold), cuando es verdadero, indica que el canal de salida analógica mantiene su valor. El valor de lectura de datos (por ejemplo, .CH2Data) muestra el valor de salida actual en unidades de ingeniería. Introduzca el tag del bit de estado InHold como el parámetro del bit InHold de la instrucción PID. Introduzca el tag del valor de lectura de datos como el parámetro del valor mantenido. Cuando el bit mantenido se hace verdadero, la instrucción PID mueve el valor mantenido en la salida de variable de control y vuelve a inicializarse para ser compatible con un reinicio sin perturbaciones en dicho valor. Cuando el módulo de salida analógica recibe su valor desde el controlador, desactiva el bit de estado InHold, lo cual permite que la instrucción PID comience a controlar normalmente.

La instrucción PID siguiente usa el bit mantenido y el valor mantenido:

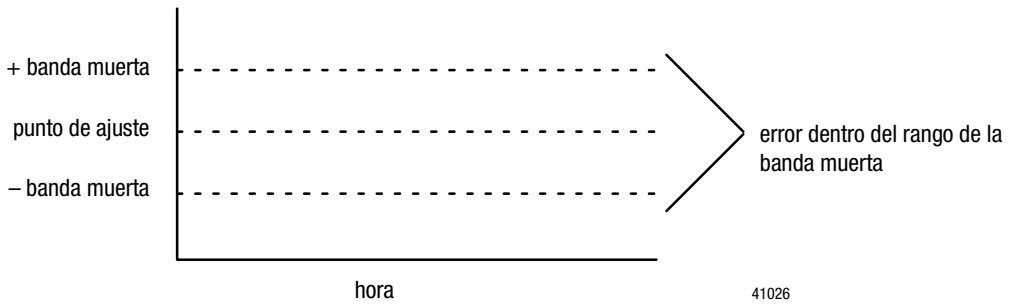


Uniformidad de derivada

El cálculo de la derivada se mejora mediante un filtro de uniformidad de derivada. Este filtro digital de primer orden y paso bajo ayuda a minimizar los picos grandes del término de derivada causados por el ruido en la PV. Esta uniformidad se hace más agresiva con los valores mayores de ganancia de derivada. Usted puede inhabilitar la uniformidad de derivada si el proceso requiere valores muy grandes de ganancia de derivada (por ejemplo, $K_d > 10$). Para inhabilitar la uniformidad de derivada, seleccione la opción "Sin uniformidad de derivada" en la ficha Configuración o establezca el bit .NDF en la estructura PID.

Cómo establecer la banda muerta

La banda muerta ajustable le permite seleccionar un rango de error por encima y por debajo del punto de ajuste donde no se cambia la salida siempre que el error permanezca dentro de este rango. Esta banda muerta le permite controlar la precisión de coincidencia de la variable de proceso con el punto de ajuste sin cambiar la salida. La banda muerta también ayuda a minimizar el desgaste del dispositivo de control final.



El paso por cero representa el control de la banda muerta que permite que la instrucción use el error para propósitos de computación a la vez que la variable de proceso cruza en la banda muerta hasta cruzar el punto de ajuste. Una vez que la variable de proceso cruza el punto de ajuste (el error cruza cero y cambia signo) y siempre que la variable de proceso permanezca en la banda muerta, no se cambiará la salida.

La banda muerta se extiende por encima y por debajo del punto de ajuste según el valor especificado. Introduzca cero para inhibir la banda muerta. La banda muerta tiene las mismas unidades de escala que el punto de ajuste. Puede usar la banda muerta sin la función de paso por cero seleccionando la opción "sin paso por cero para la banda muerta" en la ficha Configuración o establezca el bit .NOZC en la estructura PID.

Si usted usa la banda muerta, la variable de control debe ser REAL, si no se forzará a 0 cuando el error se encuentre dentro de la banda muerta

Cómo usar el límite de salida

Usted puede establecer un límite de salida (porcentaje de salida) en la salida de control. Cuando la instrucción detecta que la salida ha llegado a un límite, establece un bit de alarma e impide que la salida sobrepase el límite inferior o superior.

Prealimentación o polarización de salida

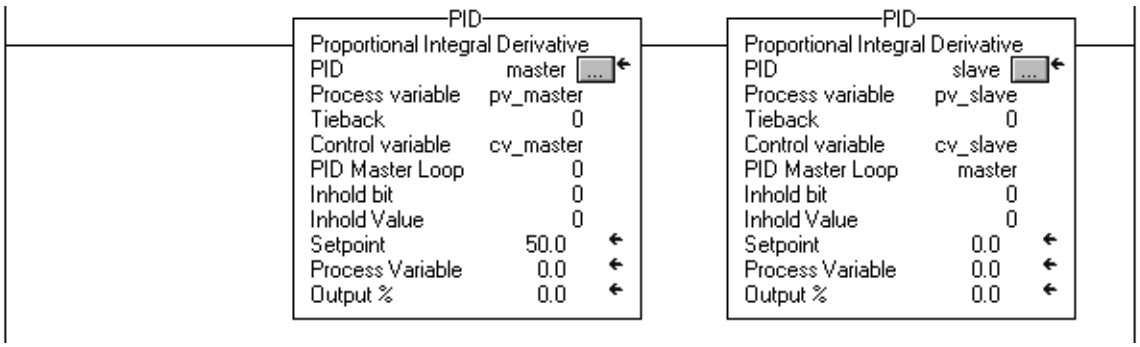
Usted puede prealimentar una perturbación del sistema alimentando el valor .BIAS en el valor de prealimentación/polarización de la instrucción PID.

El valor de prealimentación representa una perturbación alimentada en la instrucción PID antes de que la perturbación puede cambiar la variable de proceso. La prealimentación se usa frecuentemente para controlar los procesos con un retraso de transporte. Por ejemplo, un valor de prealimentación que representa “agua fría vertida en una mezcla tibia” podría reforzar el valor de salida más rápidamente que esperar hasta que cambia la variable de proceso como resultado del mezclado.

Un valor de polarización se usa típicamente cuando no se usa el control integral. En tal caso, el valor de polarización se puede ajustar para mantener la salida en el rango requerido para mantener la PV cerca del punto de ajuste.

Lazos en cascada

La instrucción PID realiza cascadas de dos lazos asignando la salida en porcentaje del lazo maestro al punto de ajuste del lazo de esclavo. El lazo de esclavo convierte automáticamente la salida del lazo maestro a las unidades de ingeniería correctas para el punto de ajuste del lazo de esclavo según los valores del lazo de esclavo para .MAXS y .MINS.



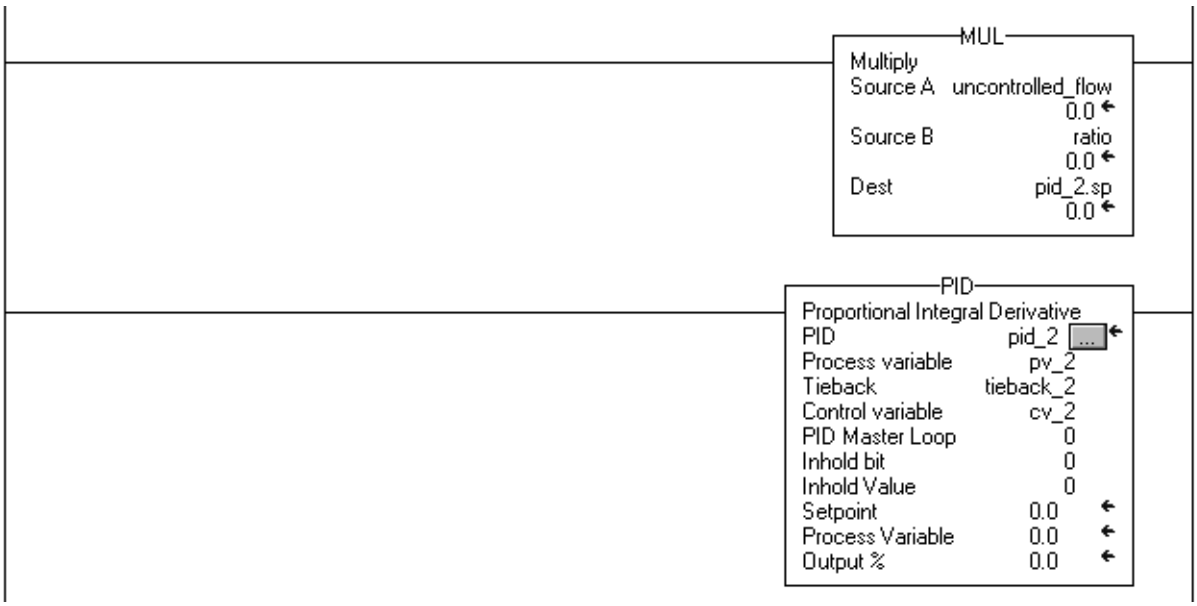
Cómo controlar una relación

Usted puede mantener dos valores en una relación usando una instrucción MUL con estos parámetros:

- valor no controlado
- valor controlado (el punto de ajuste resultante) que será usado por la instrucción PID)
- relación entre dos valores

Introduzca lo siguiente en la instrucción MUL:

Para este parámetro MUL:	Introduzca este valor:
destino	valor controlado
origen A	valor no controlado
origen B	relación



Instrucciones trigonométricas
(SIN, COS, TAN, ASN, ACS, ATN)

Introducción

Las instrucciones trigonométricas evalúan las operaciones aritméticas usando operaciones trigonométricas.

Si usted desea:	Use esta instrucción:	Vea la página:
Hallar el seno de un valor.	SIN	13-2
Hallar el coseno de un valor.	COS	13-4
Hallar la tangente de un valor.	TAN	13-6
Hallar el arco seno de un valor.	ASN	13-8
Hallar el arco coseno de un valor.	ACS	13-10
Hallar el arco tangente de un valor.	ATN	13-12

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

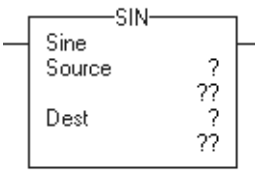
Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Una instrucción trigonométrica se ejecuta cada vez que se escanea la instrucción siempre que la condición de entrada de renglón sea verdadera. Si desea que la instrucción se evalúe solamente una vez, use una instrucción ONS para activar la instrucción trigonométrica.

Sine (SIN)

La instrucción SIN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el seno de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción SIN halla el seno del valor de origen (en radianes) y almacena el resultado en el destino.

El origen debe ser mayor o igual que –205887.4 y menor o igual que 205887.4. El valor resultante en el destino siempre es mayor o igual que –1 y menor o igual que 1.

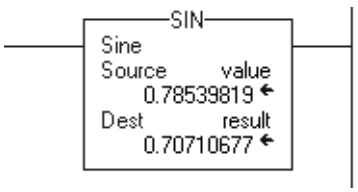
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el seno del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de SIN:



Una vez habilitada, la instrucción SIN calcula el seno de *value* y coloca el resultado en *result*.

Otros formatos:

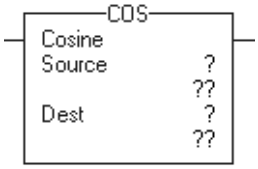
Formato:	Sintaxis:
texto neutro	`SIN(source,destination);`
texto ASCII	`SIN source destination`

Instrucciones relacionadas: CMP, CPT, COS, TAN, ASN, ACS, ATN, DEG, RAD

Cosine (COS)

La instrucción COS es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el coseno de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción COS halla el coseno del valor de origen (en radianes) y almacena el resultado en el destino.

El origen debe ser mayor o igual que –205887.4 y menor o igual que 205887.4. El valor resultante en el destino siempre es mayor o igual que –1 y menor o igual que 1.

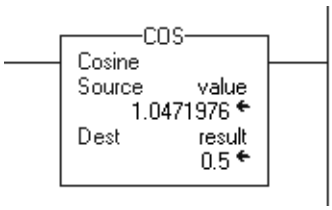
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el coseno del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de COS:



Una vez habilitada, la instrucción COS calcula el coseno de *value* y coloca el resultado en *result*.

Otros formatos:

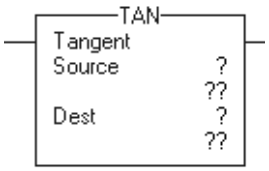
Formato:	Sintaxis:
texto neutro	`COS(source,destination);`
texto ASCII	`COS source destination`

Instrucciones relacionadas: CMP, CPT, SIN, TAN, ASN, ACS, ATN, DEG, RAD

Tangent (TAN)

La instrucción TAN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar la tangente de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción TAN halla la tangente del valor de origen (en radianes) y almacena el resultado en el destino.

El origen debe ser mayor o igual que –102943.7 y menor o igual que 102943.7.

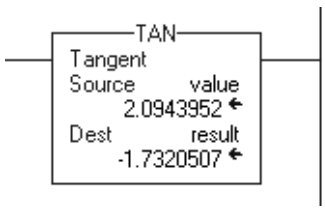
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula la tangente del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de TAN:



Una vez habilitada, la instrucción TAN calcula la tangente de *value* y coloca el resultado en *result*.

Otros formatos:

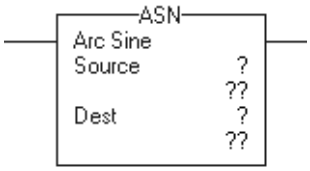
Formato:	Sintaxis:
texto neutro	`TAN(source,destination);`
texto ASCII	`TAN source destination`

Instrucciones relacionadas: CMP, CPT, COS, SIN, ASN, ACS, ATN, DEG, RAD

Arc Sine (ASN)

La instrucción ASN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el arco seno de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción ASN halla el arco seno del valor de origen y almacena el resultado en el destino (en radianes).

El origen debe ser mayor o igual que –1 y menor o igual que 1. El valor resultante en el destino siempre es mayor o igual que $-\pi/2$ y menor o igual que $\pi/2$ (donde $\pi = 3.141593$).

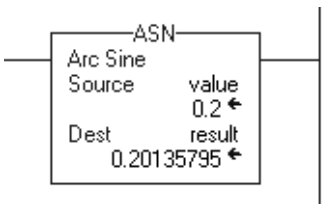
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el arco seno del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de ASN:



Una vez habilitada, la instrucción ASN calcula el arco seno de *value* y coloca el resultado en *result*.

Otros formatos:

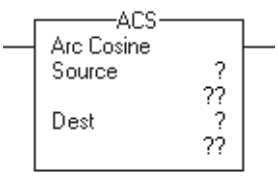
Formato:	Sintaxis:
texto neutro	`ASN(source,destination);`
texto ASCII	`ASN source destination`

Instrucciones relacionadas: CMP, CPT, ACS, ATN, SIN, COS, TAN, DEG, RAD

Arc Cosine (ACS)

La instrucción ACS es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el arco coseno de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción ACS halla el arco coseno del valor de origen y almacena el resultado en el destino (en radianes).

El origen debe ser mayor o igual que –1 y menor o igual que 1. El valor resultante en el destino siempre es mayor o igual que 0 y menor o igual que π (donde π = 3.141593).

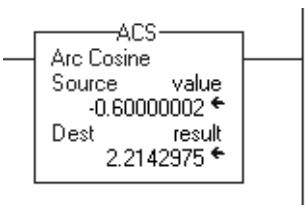
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el arco coseno del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de ASC:



Una vez habilitada, la instrucción ACS calcula el arco coseno de *value* y coloca el resultado en *result*.

Otros formatos:

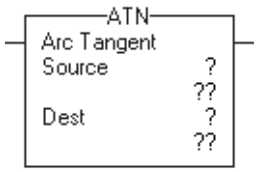
Formato:	Sintaxis:
texto neutro	`ACS(source,destination);`
texto ASCII	`ACS source destination`

Instrucciones relacionadas: CMP, CPT, ASN, ATN, SIN, COS, TAN, DEG, RAD

Arc Tangent (ATN)

La instrucción ATN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el arco tangente de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción ATN halla el arco tangente del valor de origen y almacena el resultado en el destino (en radianes).

El valor resultante en el destino siempre es mayor o igual que $-\pi/2$ y menor o igual que $\pi/2$ (donde $\pi = 3.141593$).

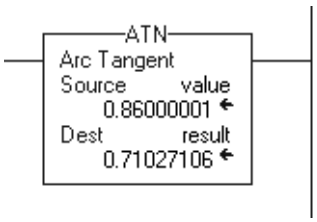
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el arco tangente del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de ATN:



Una vez habilitada, la instrucción ATN calcula el arco tangente *value* y coloca el resultado en *result*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`ATN(source,destination);`
texto ASCII	`ATN source destination`

Instrucciones relacionadas: CMP, CPT, ACS, ASN, SIN, COS, TAN, DEG, RAD

Notas:

Instrucciones matemáticas avanzadas (LN, LOG, XPY)

Introducción

Las instrucciones matemáticas avanzadas incluyen estas instrucciones:

Si usted desea:	Use esta instrucción:	Vea la página:
Hallar el logaritmo natural de un valor.	LN	14-2
Hallar el logaritmo de base 10 de un valor.	LOG	14-4
Elevar un valor a la potencia de otro valor.	XPY	14-6

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

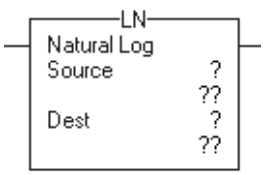
Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Una instrucción matemática avanzada se ejecuta cada vez que se escanea la instrucción siempre que la condición de entrada de renglón sea verdadera. Si desea que la instrucción se evalúe solamente una vez, use una instrucción ONS para activar la instrucción matemática.

Natural Log (LN)

La instrucción LN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el logaritmo natural de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción LN halla el logaritmo natural de origen y almacena el resultado en el destino.

El origen debe ser mayor que cero, de lo contrario se establece el bit S:V. El destino resultante es mayor o igual que –87.33655 y menor o igual que 88.72284.

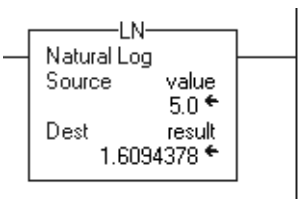
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el logaritmo natural del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de LN:



Una vez habilitada, la instrucción LN calcula el logaritmo natural de *value* y coloca el resultado en *result*.

Otros formatos:

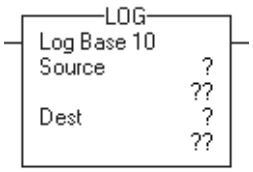
Formato:	Sintaxis:
texto neutro	`LN(source,destination);`
texto ASCII	`LN source destination`

Instrucciones relacionadas: CMP, CPT, LOG, XPY

Log Base 10 (LOG)

La instrucción LOG es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	hallar el logaritmo de este valor
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción LOG halla el logaritmo de base 10 del origen y almacena el resultado en el destino.

El origen debe ser mayor que cero, de lo contrario se establece el bit S:V. El destino resultante es mayor o igual que –37.92978 y menor o igual que 38.53184.

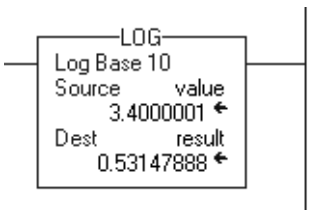
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador calcula el logaritmo del origen y coloca el resultado en el destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de LOG:



Una vez habilitada, la instrucción LOG calcula el logaritmo de *value* y coloca el resultado en *result*.

Otros formatos:

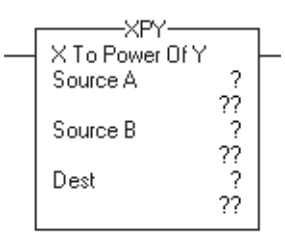
Formato:	Sintaxis:
texto neutro	`LOG(source,destination);`
texto ASCII	`LOG source destination`

Instrucciones relacionadas: CMP, CPT, LN, XPY

X to the Power of Y (XPY)

La instrucción XPY es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen A	SINT INT DINT **REAL**	valor inmediato tag	valor de base
Origen B	SINT INT DINT **REAL**	valor inmediato tag	exponente
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción eleva el origen A (X) a la potencia del origen B (Y) y almacena el resultado en el destino. Si el origen A es negativo, el origen B debe ser un valor entero de lo contrario ocurrirá un fallo menor.

La instrucción XPY usa este logaritmo: *Destino* = X**Y

El controlador evalúa $x^0 = 1$ y $0^x = 0$.

Ejecución:

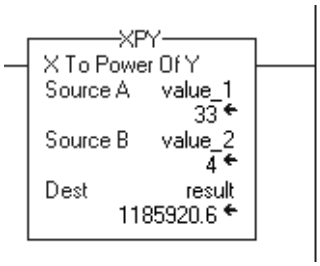
Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador eleva el origen A a la potencia del origen B y coloca el resultado en el destino. La condición de salida de renglón se establece como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo menor si:	Tipo de fallo:	Código de fallo:
El origen A es negativo y el origen B no es un valor entero	4	4

Ejemplo de XPY:



Una vez habilitada, la instrucción XPY eleva *value_1* a la potencia de *value_2* y coloca el resultado en *result*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`XPY(source_A,source_B,destination);`
texto ASCII	`XPY source_A source_B destination`

Instrucciones relacionadas: CMP, CPT, LN, LOG

Notas:

Instrucciones de conversión matemática (DEG, RAD, TOD, FRD, TRN)

Introducción

Las instrucciones de conversión matemática convierten valores.

Si usted desea:	Use esta instrucción:	Vea la página:
Convertir los radianes en grados.	DEG	15-2
Convertir los grados en radianes.	RAD	15-3
Convertir un valor de enteros a un valor BCD.	TOD	15-4
Convertir un valor BCD a un valor de enteros.	FRD	15-6
Extraer la porción fraccionaria de un valor	TRN	15-8

Usted puede combinar los tipos de datos, pero esto puede resultar en una pérdida de precisión y errores de redondeo, y la instrucción necesita más tiempo para ejecutarse. Verifique el bit S:V para ver si el resultado se truncó.

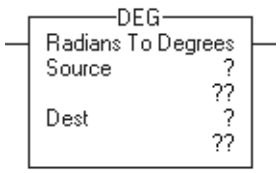
Los tipos de datos con letras **negritas** indican los tipos de datos óptimos. Una instrucción se ejecuta más rápidamente y requiere menos memoria si todos los operandos de la instrucción usan el mismo tipo de datos óptimo, típicamente DIN o REAL.

Una instrucción de conversión matemática se ejecuta cada vez que se escanea la instrucción siempre que la condición de entrada de renglón sea verdadera. Si desea que la instrucción se evalúe solamente una vez, use una instrucción ONS para activar la instrucción de conversión.

Degrees (DEG)

La instrucción DEG es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	valor que se convierte a grados
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción DEG convierte el origen (en radianes) a grados y almacena el resultado en el destino.

La instrucción RAD usa este logaritmo:
Origen*180/π (donde π = 3.141593)

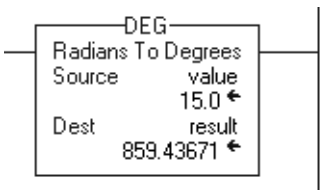
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador convierte el origen a grados y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de DEG:



Una vez habilitada, la instrucción DEG convierte *value* a grados y coloca el resultado en *result*.

Otros formatos:

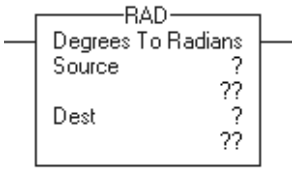
Formato:	Sintaxis:
texto neutro	`DEG(source,destination);`
texto ASCII	`DEG source destination`

Instrucciones relacionadas: CMP, CPT, RAD, SIN, COS, TAN, ASN, ACS, ATN

Radians (RAD)

La instrucción RAD es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT DINT **REAL**	valor inmediato tag	valor que se convierte a radianes
Destino	SINT INT DINT **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción RAD convierte el origen (en grados) a radianes y almacena el resultado en el destino.

La instrucción RAD usa este logaritmo:
Origen*π/180 (donde π = 3.141593)

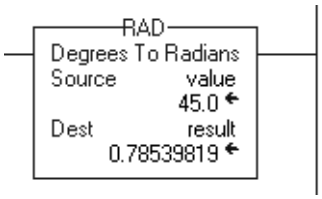
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador convierte el origen a radianes y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de RAD:



Una vez habilitada, la instrucción RAD convierte *value* a radianes y coloca el resultado en *result*.

Otros formatos:

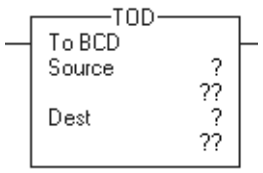
Formato:	Sintaxis:
texto neutro	`RAD(source,destination);`
texto ASCII	`RAD source destination`

Instrucciones relacionadas: CMP, CPT, DEG, SIN, COS, TAN, ASN, ACS, ATN

Convert to BCD (TOD)

La instrucción TOD es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT**	valor inmediato tag	valor que se convierte a BCD 0 ≤ Origen ≤ 99,999,999
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT**	tag	tag para almacenar el resultado

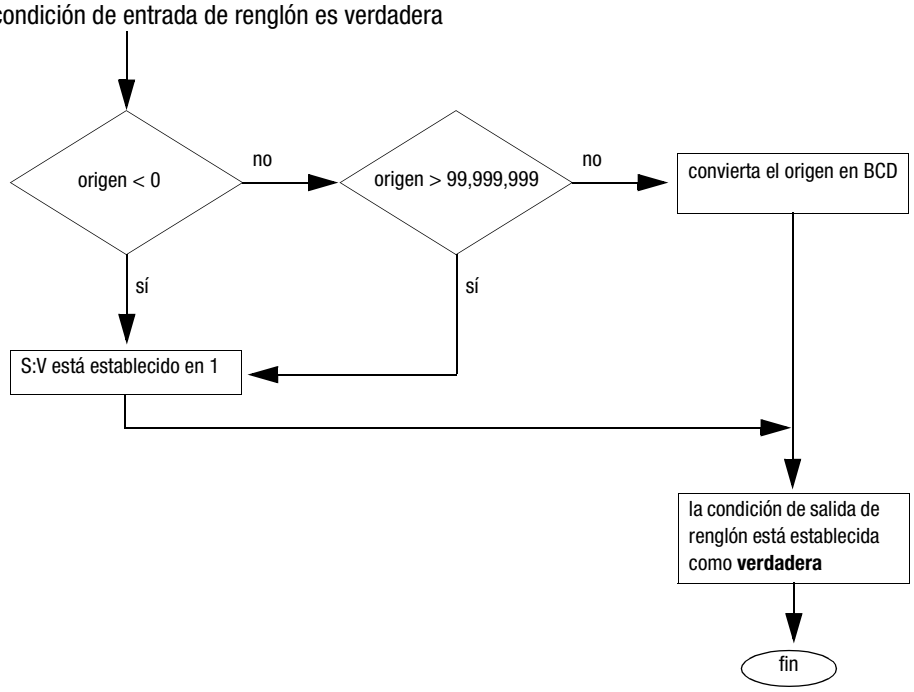
Descripción: La instrucción TOD convierte el valor decimal (0 ≤ Source ≤ 99,999,999) a un valor BCD y almacena el resultado en el destino.

El BCD representa el sistema de número decimal codificado a binario que expresa los dígitos decimales individuales (0 – 9) en una notación binaria de 4 bits.

Si usted introduce un origen negativo, la instrucción genera un fallo menor y borra el destino.

Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	



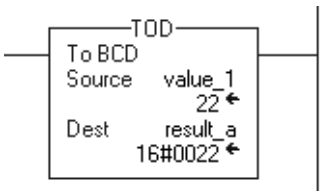
Condición:	Acción:
la condición de entrada de renglón es verdadera	El controlador convierte el origen a BCD y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo:

Ocurrirá un fallo menor si:	Tipo de fallo:	Código de fallo:
Origen < 0	4	4

Ejemplo de TOD:



Una vez habilitada, la instrucción TOD convierte *value_1* a un valor BCD y coloca el resultado en *result_a*.

Otros formatos:

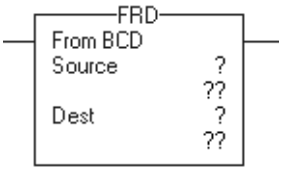
Formato:	Sintaxis:
texto neutro	`TOD(source,destination);`
texto ASCII	`TOD source destination`

Instrucciones relacionadas: CMP, CPT, FRD

Convert to Integer (FRD)

La instrucción FRD es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	SINT INT **DINT**	valor inmediato tag	valor que se convierte a decimal
	Un tag SINT o INT se convierte a un valor DINT según un relleno de cero. Consulte “Conversiones de datos” en la página A-6.		
Destino	SINT INT **DINT**	tag	tag para almacenar el resultado

Descripción: La instrucción FRD convierte un valor BCD (origen) a un valor decimal y almacena el resultado en el destino.

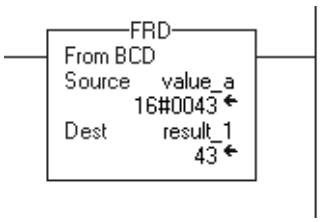
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador convierte el origen a decimal y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de FRD:



Una vez habilitada, la instrucción FRD convierte *value_a* en un valor BCD y coloca el resultado en *result_1*.

Otros formatos:

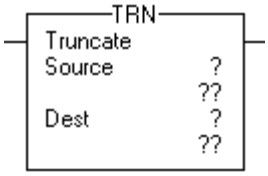
Formato:	Sintaxis:
texto neutro	`FRD(source,destination);`
texto ASCII	`FRD source destination`

Instrucciones relacionadas: CMP, CPT, TOD

Truncate (TRN)

La instrucción TRN es una instrucción de salida.

Operandos:



Operando:	Tipo:	Formato:	Descripción:
Origen	**REAL**	valor inmediato tag	valor a truncar
Destino	SINT INT **DINT** **REAL**	tag	tag para almacenar el resultado

Descripción: La instrucción TRN retira (trunca) la porción fraccionaria de un valor (origen) y almacena el resultado en el destino. El truncar no redondea el valor; la porción fraccionaria permanece la misma independientemente del valor de la porción fraccionaria.

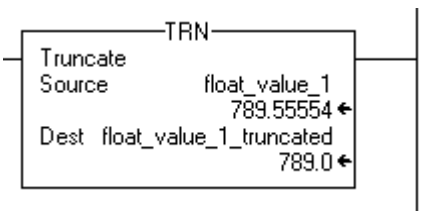
Ejecución:

Condición:	Acción:
preescán	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es falsa	La condición de salida de renglón se establece como falsa.
la condición de entrada de renglón es verdadera	El controlador retira la porción fraccionaria del origen y coloca el resultado en el destino. La condición de salida de renglón está establecida como verdadera.

Indicadores de estado aritmético: Los indicadores de estado aritmético son afectados.

Condiciones de fallo: ninguna

Ejemplo de TRN:



Cuando está habilitada, la instrucción TRN retira la porción fraccionaria de *float_value_1*, sin cambiando la porción no fraccionaria, y coloca el resultado en *float_value_1_truncated*.

Otros formatos:

Formato:	Sintaxis:
texto neutro	`TRN(source,destination);`
texto ASCII	`TRN source destination`

Instrucciones relacionadas: ABS, CMP, CPT, FRD, TOD

Notas:

Atributos comunes

Introducción

Este apéndice describe los atributos que son comunes para las instrucciones Logix5550.

Para obtener información acerca de:	Vea la página:
Palabras clave de estado aritmético	A-1
Otras palabras clave	A-4
Tipos de datos	A-5
Valores inmediatos	A-6
Conversiones de datos	A-6

Palabras clave de estado aritmético

Use las palabras clave de estado aritmético para verificar el estado de los indicadores de estado aritmético.

Palabra clave:	Indicador de estado:	Descripción:
s: V	overflow	El overflow se establece si el valor que se almacena es demasiado grande para el destino. El valor es mayor que el valor máximo para el destino o el valor es menor que el valor mínimo para el destino. **Importante:** Cada vez que s:V se cambia de restablecido a establecido, genera un fallo menor (tipo 4, código 4)
s:Z	cero	Se establece cero si el valor de destino de la instrucción es 0.
s:N	signo (el resultado es negativo)	Se establece el signo si el valor de destino de la instrucción es negativo. Vea los siguientes ejemplos.
s:C	acarreo	Se establece el acarreo si una operación aritmética causa un “acarreo” o un “acarreo negativo” que intenta usar los bits fuera del tipo de datos de destino. Por ejemplo: sumar 3 y 9 causa un valor de acarreo de 1 restar 25 de 18 requiere un acarreo negativo de 10 Vea los siguientes ejemplos.

Las palabras clave de estado aritmético pueden utilizar mayúsculas/minúsculas.

Puesto que los indicadores de estado aritmético pueden cambiar rápidamente, el software no proporciona una representación animada del tag asociado. Como consecuencia, se puede determinar visualmente el estado de los indicadores.

Los diagramas siguientes indican dónde los tipos de datos enteros almacenan los indicadores de estado S:N y S:C.

Si el tipo de datos es SINT

valor máximo = +127
valor mínimo = –128

s:C	7 s:N	6	5	4	3	2	1	0

Si el tipo de datos es INT

valor máximo = +32,767
valor mínimo = –32,768

s:C	15 s:N	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

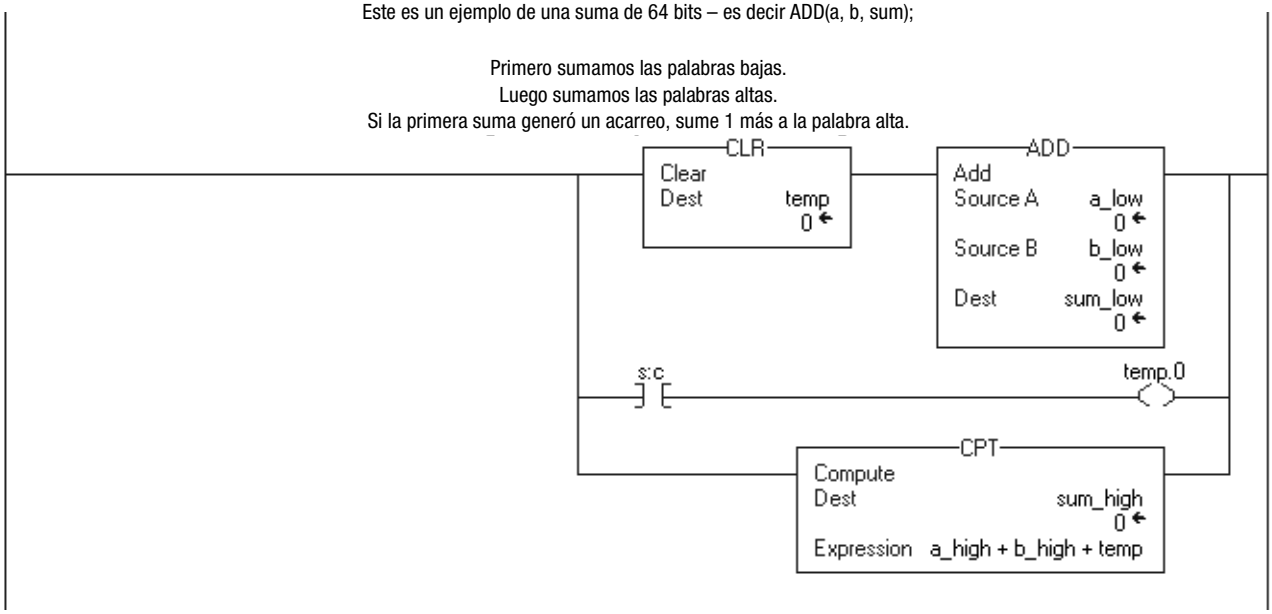
Si el tipo de datos es DINT

valor máximo = +2,147,483,647
valor mínimo = –2,147, 483,648

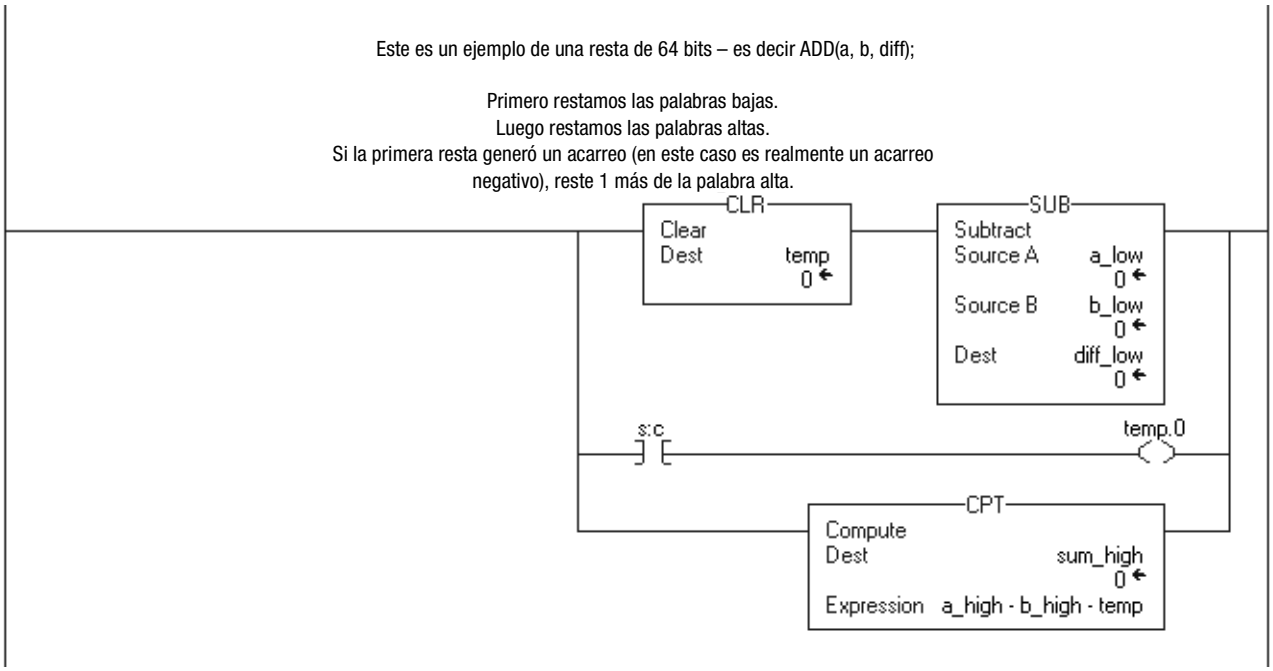
s:C	31 s:N	30	29	28	27	26	25	24	• • • • •	7	6	5	4	3	2	1	0

Los siguientes ejemplos de programación le indican cómo usted puede usar el bit de acarreo.

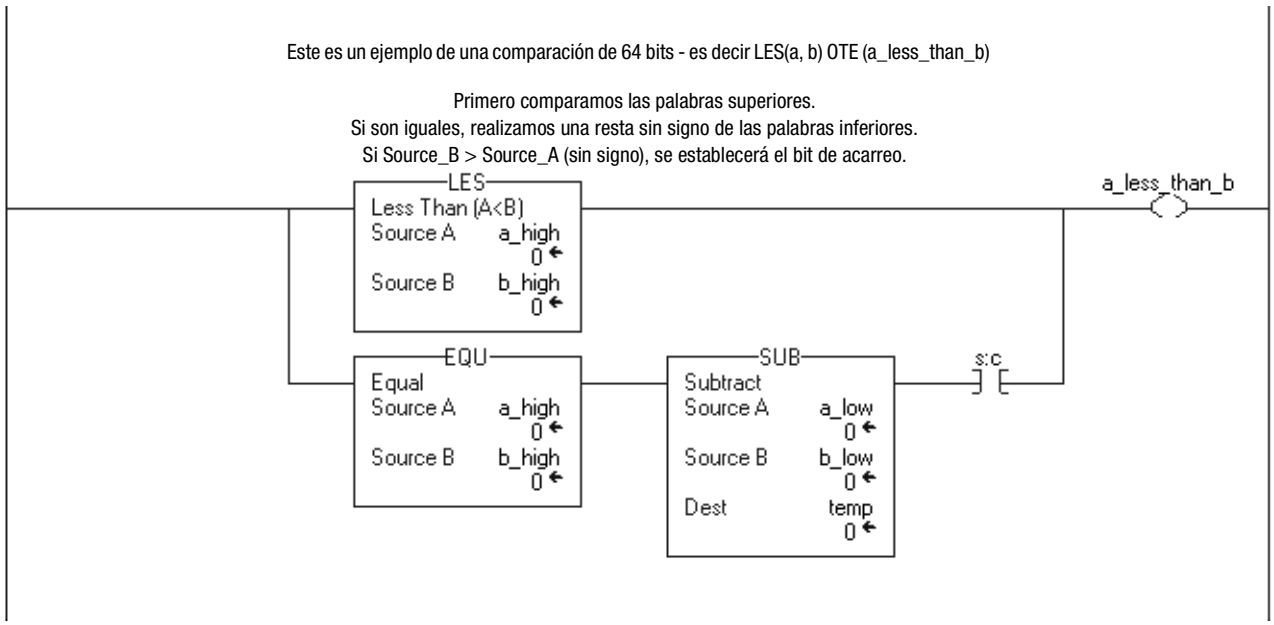
ejemplo 1



ejemplo 2



ejemplo 3


Este es un ejemplo de una comparación de 64 bits - es decir LES(a, b) OTE (a_less_than_b)
Primero comparamos las palabras superiores.
Si son iguales, realizamos una resta sin signo de las palabras inferiores.
Si Source_B > Source_A (sin signo), se establecerá el bit de acarreo.
LES
Less Than (A<B)
Source A a_high 0
Source B b_high 0
a_less_than_b
EQU
Equal
Source A a_high 0
Source B b_high 0
SUB
Subtract
Source A a_low 0
Source B b_low 0
Dest temp 0
s:c

Otras palabras clave

Además de las palabras clave de estado aritmético, el controlador es compatible con las siguientes palabras clave.

Palabra clave:	Acceso:	Descripción:
s:FS	lectura	El bit de primer escán se establece si éste es el primer escán Normal de las rutinas en el programa actual.
s:MINOR	lectura escritura	El bit de fallo menor se establece si se ha generado por lo menos un fallo menor. El controlador establece este bit cuando ocurre un fallo menor debido a la ejecución del programa. El controlador no establece este bit para los fallos menores que no estén asociados con la ejecución del programa; por ejemplo, batería débil.
Esto:	n/a	La instrucción THIS es válida solamente con las instrucciones GSV y SSV que se refieren a TASK, PROGRAM o ROUTINE. Use la instrucción THIS para especificar las instrucciones TASK, PROGRAM o ROUTINE actuales.

Las palabras clave de estado pueden utilizar mayúsculas/minúsculas.

Puesto que estos indicadores de estado cambian rápidamente, estas palabras clave de estado no tienen una representación animada en el software de programación para mostrar el estado real.

Usted **no puede** definir un alias de tag para una palabra clave.

Tipos de datos

Los tipos de datos del controlador cumplen con las especificaciones de los tipos de datos definidos en IEC 1131-3. Los tipos de datos atómicos predefinidos son:

Tipos de datos:	Descripción:	Rango:
BOOL	booleano de 1 bit	0 = restablecido 1 = establecido
SINT	número entero de 1 byte	–128 a 127
INT	número entero de 2 bytes	–32,768 a 32,767
DINT	número entero de 4 bytes	–2,147,483,648 a 2,147,483,647
real	número de punto flotante (coma flotante) de 4 bytes	$-3.402823E^{38}$ a $-1.1754944E^{-38}$ (valores negativos) y 0 y $1.1754944E^{-38}$ a $3.402823E^{38}$ (valores positivos)

El tipo de datos REAL también almacena ±infinito y ±NAN, pero la pantalla del software es diferente según el formato de la pantalla.

Formato de la pantalla:	Equivalencia:	
Real	+ infinito	1.$
	– infinito	–1.$
	+ NAN	1.#QNAN
	– NAN	–1.#QNAN
Exponencial	+ infinito	1.#INF000e+000
	– infinito	–1.#INF000e+000
	+NAN	1.#QNAN00e+000
	+NAN	–1.#QNAN00e+000

Las estructuras predefinidas son:

Tipos de datos:	Descripción:
AXIS(1)	estructura de control para un eje
CONTROL	estructura de control para las instrucciones de arreglo (archivo)
COUNTER	estructura de control para las instrucciones del contador
MESSAGE(1)	estructura de control para la instrucción MSG
MOTION_GROUP(1)	estructura de control para un grupo de movimiento
MOTION_INSTRUCTION	estructura de control para las instrucciones de movimiento
PID	estructura de control para las instrucciones PID
TIMER	estructura de control para las instrucciones del temporizador

(1) Estas estructuras no son compatibles con arreglos, no se pueden anidar en estructuras definidas por el usuario ni se pueden pasar a otras rutinas mediante una instrucción JSR.

Valores inmediatos

Cuando se introduce un valor inmediato (constante) en formato decimal (es decir, –2, 3), el controlador almacena el valor usando 32 bits. Si se introduce un valor en una base que no es decimal, tal como binario o hexadecimal, y no se especifican los 32 bits, el controlador coloca un cero en los bits que no se especifican (relleno de cero).

Ejemplo: Relleno con cero de valores inmediatos

Si se introduce:	El controlador almacena:
–1	16#ffff ffff (–1)
16#ffff (-1)	16#0000 ffff (65535)
8#1234 (668)	16#0000 029c (668)
2#1010 (10)	16#0000 000a (10)

Conversiones de datos

Las instrucciones Logix5000 se ejecutan más rápidamente y requieren menos memoria si todos los operandos de la instrucción usan:

- el mismo tipo de datos
- un tipo de datos óptimo:
  - En la sección “Operandos” de cada instrucción que aparece en este manual, un tipo de datos con letras **negritas** indica un tipo de datos óptimo.
  - Los tipos de datos DINT y REAL son generalmente los tipos de datos óptimos.

Si se combinan tipos de datos y usan tags que no son del tipo de datos óptimo, el controlador convierte los datos según estas reglas

- ¿Son *cualquiera* de los operandos un valor REAL?

Si:	Los operandos de entrada (es decir, origen, tag en una expresión, límite) se convierten a:
Sí	REAL
No	DINT

- Después de la ejecución de la instrucción, el resultado (un valor DINT o REAL) se convierte al tipo de datos de destino, si fuera necesario.

▶ Actualmente no es posible especificar un tag BOOL en una instrucción que realiza una operación en los tipos de datos enteros o REAL.

Puesto que la conversión de datos requiere tiempo y memoria adicionales, es posible mejorar la eficacia de los programas haciendo lo siguiente:

- usar el mismo tipo de datos en toda la instrucción
- minimizar el uso de los tipos de datos SINT o INT

Es decir, use todos los tags DINT o todos los tags REAL además de los valores inmediatos en las instrucciones.

Las secciones siguientes explican cómo se convierten los datos cuando se usan los tags SINT o INT o cuando se combinan tipos de datos.

SINT o INT a DINT

Para las instrucciones que convierten los valores SINT o INT a valores DINT, las secciones “Operandos” en este manual indican el método de conversión.

Este método de conversión:	Convierte los datos colocando:
Extensión con signo	el valor en el bit del extremo izquierdo (el signo del valor) en cada posición de bit a la izquierda de los bits existentes hasta que haya 32 bits.
Relleno con ceros	ceros a la izquierda de los bits existentes hasta que haya 32 bits

El siguiente ejemplo muestra los resultados de convertir un valor usando la extensión con signo y el relleno con ceros.

Este valor	2#1111_1111_1111_1111	(-1)
Se convierte a este valor según la extensión con signo	2#1111_1111_1111_1111_1111_1111_1111_1111	(-1)
Se convierte a este valor según el relleno con ceros	2#0000_0000_0000_0000_1111_1111_1111_1111	(65535)

Puesto que los valores inmediatos son siempre rellenos con cero, la conversión de un valor SINT o INT *puede* producir resultados inesperados. En el ejemplo siguiente, la comparación es falsa porque el origen A, un INT, se convierte según la extensión con signo; en cambio, el origen B, un valor inmediato, tiene un relleno con ceros.

Si se usan un tag SINT o INT y un valor inmediato en una instrucción que convierte los datos según la extensión con signo, use uno de los siguientes métodos para manejar los valores inmediatos:

- Especifique el valor inmediato en la base decimal
- Si introduce un valor en una base que no es decimal, especifique los 32 bits del valor inmediato. Para hacerlo, introduzca el valor del bit del extremo izquierdo en cada posición de bit a la izquierda hasta que haya 32 bits.
- Cree un tag para cada operando y use el mismo tipo de datos por toda la instrucción. Para asignar un valor constante, haga lo siguiente:
  - Introdúzcalo en uno de los tags, o bien
  - Añada una instrucción MOV que transfiere el valor en uno de los tags.
- Use una instrucción MEQ para verificar solamente los bits requeridos.

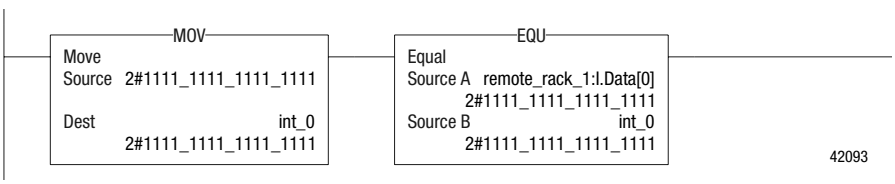
Los siguientes ejemplos muestras dos maneras de combinar un valor inmediato y un tag INT. Los dos ejemplos verifican los bits de un módulo de E/S 1771 para determinar si los bit están establecidos. Puesto que la palabra de datos de un módulo de E/S 1771 es un tag INT, es más conveniente usar un valor constante de 16 bits.

Ejemplo 1: Combinar un tag INT y un valor inmediato

Puesto que *remote_rack_1:I. Data[0]* es un tag INT, el valor con el cual comprobarlo también se introduce como tag INT.

Ejemplo 2: Combinar un tag INT y un valor inmediato

Puesto que *remote_rack_1:I. Data[0]* es un tag INT, el valor con el cual comprobarlo se transfiere primero a *int_0*, el cual también es un tag INT. La instrucción EQU compara los dos tags.



Número entero a REAL

El controlador almacena los valores REAL en un formato numérico de punto flotante (coma flotante) de precisión única IEEE. Este usa un bit para el signo del valor, 23 bits para el valor de base y ocho bits para la exponente (total de 32 bits). Si se combinan un tag entero (SINT, INT o DINT) y un tag REAL como entradas en la misma instrucción, el controlador convierte el valor entero a un valor REAL antes de ejecutarse la instrucción.

- Un valor SINT o INT siempre se convierte al mismo valor REAL.
- Es posible que un valor no se convierta al mismo valor REAL:
  - Un valor REAL usa hasta 24 bits para el valor de base (23 bits almacenados más un bit “oculto”).
  - Un valor DINT usa hasta 32 bits para el valor (uno para el signo y 32 para el valor)
  - Si el valor DINT requiere más de 24 bits significativos, es posible que *no* convierta al mismo valor REAL. En este caso, el controlador redondea el valor al valor REAL más cercano usando 24 bits significativos.

DINT a SINT o INT

Para convertir un valor DINT a un valor SINT o INT, el controlador trunca la porción superior del DINT y establece el indicador de estado de overflow, si fuera necesario. El siguiente ejemplo muestra el resultado de una conversión de DINT a SINT o INT.

Ejemplo: Conversión de un DINT a INT y SINT

Este valor DINT:	Se convierte a este valor menor:	
16#0001_0081 (65,665)	INT:	16#0081 (129)
	SINT:	16#81 (-127)

REAL a un número entero

Para convertir un valor REAL a un valor entero, el controlador redondea la porción fraccionaria y trunca la porción superior de la porción no fraccionaria. Si se pierden los datos, el controlador establece el indicador de estado de overflow. Los números se redondean de la manera siguiente:

- Los números que no son *x*.5 se redondean al número entero más cercando.
- *X*.5 se redondea al número par más cercano.

El siguiente ejemplo muestra el resultado de convertir los valores REAL a valores DINT.

Ejemplo: Conversión de valores REAL a valores DINT

Este valor REAL:	Se convierte a este valor DINT:
–2.5	–2
–1.6	–2
–1.5	–2
–1.4	–1
1.4	1
1.5	2
1.6	2
2.5	2

Importante: Los indicadores de estado aritmético se establecen según el valor almacenado. Las instrucciones que normalmente no afectan las palabras clave de estado aritmético pueden simular este efecto si la conversión del tipo ocurre debido a tipos de datos combinados para los parámetros de instrucción. El proceso de conversión del tipo establece las palabras clave de estado aritmético.

Conceptos de arreglo

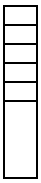
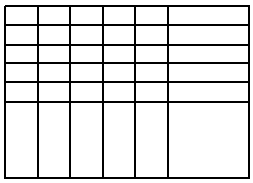
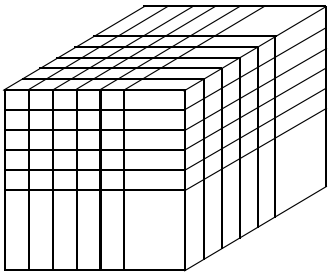
Cómo ver un arreglo como colección de elementos

Los arreglos le permiten agrupar un conjunto de datos (del mismo tipo de datos) según el mismo nombre y usar subíndices para identificar los elemento individuales. Un elemento en un arreglo puede ser un tipo o estructura de datos atómicos.

Usted especifica un elemento en un arreglo según sus subíndices. Introduzca el nombre de tag del arreglo seguido por los subíndices entre corchetes. Los subíndices deben especificar un valor para cada dimensión del arreglo. Las dimensiones están basadas en cero.

Para este arreglo:	**Especifique:**
una dimensión	*array_name[subscript_0]*
dos dimensiones	*array_name[subscript_0, subscript_1]*
tres dimensiones	*array_name[subscript_0, subscript_1, subscript_2]*

Un arreglo puede tener hasta tres dimensiones. El número total de elementos en un arreglo es el producto del tamaño de cada dimensión.

Este arreglo:	**Almacena los datos como:**	**Por ejemplo:**
una dimensión		**Nombre de tag:** **Tipo** **Dimensión 0** **Dimensión 1** **Dimensión 2** *one_d_array* DINT[7] 7 – – número total de elementos = 7 rango de subíndice válido DINT[x] donde x = 0 – 6
dos dimensiones		**Nombre de tag:** **Tipo** **Dimensión 0** **Dimensión 1** **Dimensión 2** *two_d_array* DINT[4,5] 4 5 – número total de elementos = 4 ∗ 5 = 20 rango de subíndice válido DINT[x, Y] donde x = 0 – 3; Y = 0 – 4
tres dimensiones		**Nombre de tag:** **Tipo** **Dimensión 0** **Dimensión 1** **Dimensión 2** *three_d_array* DINT[2,3,4] 2 3 4 número total de elementos = 2 ∗ 3 ∗ 4 = 24 rango de subíndice válido DINT[x, Y, z] donde x = 0 – 1; Y = 0 – 2, z = 0 – 3

Cómo indexar a través de los arreglos

Para cambiar dinámicamente el elemento de arreglo al cual se refiere la lógica, use el tag o expresión como el subíndice para indicar el elemento. Esto es similar al direccionamiento indirecto en la lógica PLC-5. Puede usar estos operadores en una expresión para especificar un subíndice de arreglo:

Operador:	Descripción:
+	sumar
–	restar/cambiar signo
*	multiplicar
/	dividir
ABS	Valor absoluto
AND	AND
FRD	BCD a número entero
MOD	Módulo
NOT	complemento
OR	OR
SQR	raíz cuadrada
TOD	número entero a BCD
TRUN	Truncar
XOR	O exclusivo

Por ejemplo:

Definiciones:		Ejemplo:	Descripción:
my_list	definido como DINT[10]	*my_list[5]*	Este ejemplo se refiere al elemento 5 en el arreglo. La referencia es estática porque el valor de subíndice permanece constante.
my_list *position*	definido como DINT[10] definido como DINT	use la instrucción MOV para poner el valor 5 en *position* *my_list[position]*	Este ejemplo se refiere al elemento 5 en el arreglo. La referencia es dinámica porque la lógica puede cambiar el subíndice cambiando el valor de *position*.
my_list *position* *offset*	definido como DINT[10] definido como DINT definido como DINT	use la instrucción MOV para poner el valor 2 en *position* use la instrucción MOV para poner el valor 5 en *offset* *my_list[position + offset]*	Este ejemplo se refiere al elemento 7 (2 + 5) en el arreglo. La referencia es dinámica porque la lógica puede cambiar el subíndice cambiando el valor de *position* u *offset*.

Asegúrese de que cualquier subíndice de arreglo que introduce se encuentre dentro de los límites del arreglo especificado. Las instrucciones que ven los arreglos como un conjunto de elementos generan un fallo mayor (tipo 4, código 20) si un subíndice excede su dimensión correspondiente.

Cómo especificar bit dentro de arreglos

Puede direccionar los bits dentro de elementos de arreglos. Por ejemplo:

Definiciones:		Ejemplo:	Descripción:
array1	definido como DINT[5]	*array1[1].2*	Este ejemplo se refiere al bit 2 en el elemento 1 del arreglo.
array2	definido como INT[17,36] la 1ª dimensión tiene 17 elementos la 2ª dimensión tiene 36 elementos	*array2 [3, 4].15*	Este ejemplo se refiere al bit 15 del elemento *array2[3,4]*.
array3	definido como SINT[2,4,6] la 1ª dimensión retiene 2 elementos la 2ª dimensión retiene 4 elementos la 3ª dimensión retiene 6 elementos	*array3 [1, 3, 2].4*	Este ejemplo se refiere al bit 4 del elemento *array3 [1, 3, 2]*.
MyArray *MyIndex*	definido como SINT[100] definido como SINT	*MyArray[(MyIndex AND NOT* 7) / 8]. [MyIndex AND 7]	Este ejemplo se refiere a un bit dentro de un arreglo SINT.
MyArray *MyIndex*	definido como INT[100] definido como INT	*MyArray[(MyIndex AND NOT* 15) / 16]. [MyIndex AND 15]	Este ejemplo se refiere a un bit dentro de un arreglo INT.
MyArray *MyIndex*	definido como DINT[100] definido como DINT	*MyArray[(MyIndex AND NOT* 31) / 32]. [MyIndex AND 31]	Este ejemplo se refiere a un bit dentro de un arreglo DINT.

También puede usar los operadores indicados en la tabulación de la página B-3 para especificar los bits.

Cómo ver un arreglo como un bloque de memoria

Los datos en un arreglo se almacenan contiguamente en la memoria. Las instrucciones de archivo (arreglo) típicamente requieren una dirección inicial dentro de un arreglo y una longitud, la cual determina qué elementos y cuántos elementos la instrucción lee o escribe.

Importante: Si una instrucción intenta leer datos más allá del fin de un arreglo, la instrucción lee los datos que hay y los procesa como si fueran datos válidos (no ocurre un error). Si una instrucción intenta escribir datos más allá del fin de un arreglo, ocurre un fallo mayor (tipo 4, código 20).

Las instrucciones siguientes manipulan los datos de arreglo como un bloque continuo de memoria (las demás instrucciones manipulan los datos de arreglo como elementos individuales):

BSL	FBC	LFL	SQO
BSR	FFL	LFU	
COP	FFU	SQI	
DDT	FLL	SQL	

Cómo el controlador almacena los datos del arreglo

La tabla siguiente indica el orden secuencial de los elementos en los ejemplos de la página B-2.

Elementos del arreglo de una dimensión (orden ascendente):	**Elementos del arreglo de dos dimensiones (orden ascendente):**	**Elementos del arreglo de tres dimensiones (orden ascendente):**
one_D_array[0] one_D_array[1] one_D_array[2] one_D_array[3] one_D_array[4] one_D_array[5] one_D_array[6] Para un arreglo con solamente una dimensión, *tag_name[subscript_0]*, *subscript_0* se incrementa a su valor máximo.	two_D_array[0,0] two_D_array[0,1] two_D_array[0,2] two_D_array[0,3] two_D_array[0,4] two_D_array[1,0] two_D_array[1,1] two_D_array[1,2] two_D_array[1,3] two_D_array[1,4] two_D_array[2,0] two_D_array[2,1] two_D_array[2,2] two_D_array[2,3] two_D_array[2,4] two_D_array[3,0] two_D_array[3,1] two_D_array[3,2] two_D_array[3,3] two_D_array[3,4] Para un arreglo con dos dimensiones, *tag_name[subscript_0, subscript_1]*, *subscript_0* se mantiene fijo en 0 mientras *subscript_1* se incrementa de 0 a su valor máximo. *Subscript_0* se incrementa por 1 (si dimensión 0 es mayor que 1) y se mantiene fijo mientras *subscript_1* se incrementa nuevamente a través de su rango. Este proceso continúa hasta que los dos subíndices llegan a sus valores máximos.	three_D_array[0,0,0] three_D_array[0,0,1] three_D_array[0,0,2] three_D_array[0,0,3] three_D_array[0,1,0] three_D_array[0,1,1] three_D_array[0,1,2] three_D_array[0,1,3] three_D_array[0,2,0] three_D_array[0,2,1] three_D_array[0,2,2] three_D_array[0,2,3] three_D_array[1,0,0] three_D_array[1,0,2] three_D_array[1,0,3] three_D_array[1,1,0] three_D_array[1,1,1] three_D_array[1,1,2] three_D_array[1,1,3] three_D_array[1,2,0] three_D_array[1,2,1] three_D_array[1,2,2] three_D_array[1,2,3] Para un arreglo con tres dimensiones, *tag_name[subscript_0, subscript_1, subscript_2]*, *subscript_0* se mantiene fijo en 0 mientras *subscript_1* y *subscript_2* se incrementan como un arreglo de dos dimensiones. *subscript_0* se incrementa por 1 (si la dimensión 0 es mayor que 1) y se mantiene fijo hasta que *subscript_1* y *subscript_2* llegan a sus valores máximos. Este proceso continúa hasta que los tres subíndices llegan a sus valores máximos.

Cómo variar una dimensión

Las instrucciones AVE, SRT y STD tienen una dimensión para variar el operando. La instrucción usa este operando para calcular un offset que la instrucción usa para determinar qué elementos del arreglo leer o escribir.

Arreglo:	Dimensión para variar:	offset:
una dimensión	0	1
dos dimensiones	0	dimension_1
	1	1
tres dimensiones	0	(*dimension_1*) ∗ (*dimension_2*)
	1	dimension_2
	2	1

Asignación de memoria para los arreglos

La cantidad de memoria que un arreglo usa depende del tipo de datos usado para crear el arreglo. La asignación mínima dentro del controlador es cuatro bytes, lo cual es igual a 32 BOOL, 4 SINT, 2 INT ó 1 DINT.

Los ejemplos siguientes indican la asignación de memoria para varios arreglos:

bit_values como BOOL[32] Este ejemplo es un arreglo con elementos de 32 bits, cada uno del tipo de datos BOOL (1 bit por elemento).

bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
asignación de datos 1	[15]	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
bit:	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
asignación de bit 1 *continúa*	[31]	[30]	[29]	[28]	[27]	[26]	[25]	[24]	[23]	[22]	[21]	[20]	[19]	[18]	[17]	[16]

Este ejemplo usa 64 bits de memoria.

small_values como SINT[8] Este ejemplo representa un arreglo con 8 elementos, cada uno del tipo de datos SINT (1 byte para cada elemento).

bit:	31 24	23 16	15 8	7 0
asignación de datos 1	small_values[3]	small_values[2]	small_values[1]	small_values[0]
asignación de datos 2	small_values[7]	small_values[6]	small_values[5]	small_values[4]

Este ejemplo usa 64 bits de memoria.

small_values como SINT[3] Este ejemplo es un arreglo con 3 elementos, cada uno del tipo de datos SINT (1 byte para cada elemento). Puesto que la asignación de datos mínima es 4 bytes, el último byte es cero.

bit:	31 24	23 16	15 8	7 0
asignación de datos 1	0	small_values[2]	small_values[1]	small_values[0]

Este ejemplo usa 32 bits de memoria.

valores como INT[4] Este ejemplo es un arreglo con 4 elementos, cada uno del tipo de datos INT (2 bytes para cada elemento).

bit:	31 16	15 0
asignación de datos 1	values[1]	values[0]
asignación de datos 2	values[3]	values[2]

Este ejemplo usa 64 bits de memoria.

big_values como DINT[2] Este ejemplo es un arreglo con 2 elementos, cada uno del tipo de datos DINT (4 bytes para cada elemento).

bit:	31 0
asignación de datos 1	big_values[0]
asignación de datos 2	big_values[1]

Este ejemplo usa 64 bits de memoria.

timer_list como TIMER[2] Este ejemplo es un arreglo con 2 elementos, cada uno del tipo de datos TIMER (12 bytes para cada estructura).

bit:	31 0
asignación de datos 1	bits de estado timer_list[0]
asignación de datos 2	timer_list[0].pre
asignación de datos 3	timer_list[0].acc
asignación de datos 4	bits de estado timer_list[1]
asignación de datos 5	timer_list[1].pre
asignación de datos 6	timer_list[1].acc

Este ejemplo usa 192 bits de memoria.

small_values como SINT[2,2,2] Este ejemplo es un arreglo de tres dimensiones con 8 elementos, cada uno del tipo de datos SINT.

bit:	31 24	23 16	15 8	7 0
asignación de datos 1	small_values [0,1,1]	small_values [0,1,0]	small_values [0,0,1]	small_values [0,0,0]
asignación de datos 2	small_values [1,1,1]	small_values [1,1,0]	small_values [1,0,1]	small_values [1,0,0]

Este ejemplo usa 64 bits de memoria.

big_values como DINT[2,2,2] Este ejemplo es un arreglo de tres dimensiones con 8 elementos, cada uno del tipo de datos DINT.

bit:	31 0
asignación de datos 1	big_values[0,0,0]
asignación de datos 2	big_values[0,0,1]
asignación de datos 3	big_values[0,1,0]
asignación de datos 4	big_values[0,1,1]
asignación de datos 5	big_values[1,0,0]
asignación de datos 6	big_values[1,0,1]
asignación de datos 7	big_values[1,1,0]
asignación de datos 8	big_values[1,1,1]

Este ejemplo usa 256 bits de memoria.

Puede modificar las dimensiones del arreglo cuando programa fuera de línea sin perder los datos del tag. No puede modificar las dimensiones del arreglo cuando programa en línea.

Tiempo de ejecución

Introducción

Use este apéndice para calcular el tiempo requerido para la ejecución de una instrucción. Los tiempos son para un controlador Logix5550 (No. de cat. 1756-L1) y el software RSLogix 5000, versión 2.25.

Importante: Use esta información para seleccionar las varias opciones de programación. Los tiempos de ejecución reales pueden variarse debido a la configuración del proyecto y la versión del software RSLogix 5000 que se usa.

Para calcular el tiempo de ejecución de una instrucción:

1. En la Table C.2 en la página C-3, halle el tiempo de base para la instrucción.

2. Añada tiempo al tiempo de base de la expresión para expresiones, subíndices y conversiones de datos:

Si un operando:	Entonces añada tiempo para:	Usando esta tabla:
contiene una expresión	cada operación en la expresión. Para cada operación, use el tiempo para la instrucción correspondiente.	Table C.2 en la página C-3
es un elemento de un arreglo y uno o más de los subíndices es un tag (por ejemplo, *tag_c[tag_d]*)	el cálculo del subíndice de arreglo	Table C.5 en la página C-15
es convertido por una extensión con signo	la conversión de datos. Consulte “Conversiones de datos” en la página A-6.	Table C.1 en la página C-3
contiene un tag SINT o INT en un subíndice de arreglo	la conversión del SINT o INT a un DINT	

Ejemplos:

Instrucción:	Tipo de datos:	Tiempo de ejecución:
CMP (*tag_a * tag_b/100)*		tiempo de procesamiento interno de 4.3 µs
* (MUL)	REAL	17.7 µs para la operación MUL
/(DIV)	REAL	18.3 µs para la operación DIV
		total de 40.3 µs
ADD		10.7 µs para la instrucción ADD
Origen A	DINT	8.4 µs para la conversión de DINT a REAL
Origen B	REAL	
Destino	REAL	
		total de 19.1 µs
ADD		10.7 µs para la instrucción ADD
Origen A	DINT[0,DINT]	25.5 µs para el subíndice de arreglo 8.4 µs la para conversión de DINT a REAL de un elemento de arreglo
Origen B	REAL	
Destino	REAL	
		total de 44.6 µs
MUL		17.7 µs para la instrucción MUL
Origen A	REAL	
Origen B	DINT	8.4 µs para la conversión de DINT a REAL
Destino	DINT	12.3 µs para la conversión de REAL a DINT
		total de 38.4 µs

Tablas de referencia

Table C.1 Tiempos de conversión de extensión con signo

Para convertir un:	A un:	Requiere (µs):
SINT	INT	8.3
	DINT	0.98
	REAL	2.1
INT	SINT	8.3
	DINT	1.1
	REAL	2.3
DINT	SINT	7.3
	INT	7.4
	REAL	8.4
REAL	SINT	14.7
	INT	14.8
	DINT	12.3

Table C.2 Tiempos de ejecución de instrucción

Instrucción:	Tipo de datos óptimo:	Tiempo verdadero (µs):	Tiempo falso (µs):	Notas:
ABS	DINT	0.59	0.17	
	REAL	0.89	0.19	
ACS	REAL	268.2	0.12	
ADD	DINT	0.56	0.10	
	REAL	10.7	0.11	
AFI	na	0.05		
AND	DINT	0.56	0.11	
ASN	REAL	258.9	0.11	
ATN	REAL	220.9	0.11	
AVE	REAL	43.5 + (longitud * 14.51)	23.5	
BRK	na	0		
BSL	DINT	7.8 + ((longitud/32) * 0.77)	4.3	Longitud de redondear hacia arriba/32 a un número entero.
BSR	DINT	8.7 + ((longitud/32) * 0.77)	4.3	Longitud de redondear hacia arriba/32 a un número entero.
BTD	DINT	3.3	0.16	

Table C.2 Tiempos de ejecución de instrucción (Continued)

Instrucción:	Tipo de datos óptimo:	Tiempo verdadero (µs):	Tiempo falso (µs):	Notas:
CLR	DINT	0.35	0.10	
	REAL	0.40	0.11	
CMP	Refer to the operations within the expression.	4.3 + *expresión*	0.11	
COP	SINT	5.8 + (longitud * 0.09)		
	INT	5.6 + (longitud * 0.17)		
	DINT, REAL	6.0 + (longitud * 0.35)	0.15	
COS	REAL	218.7	0.08	
CPT	Refer to the operations within the expression.	4.7 + *expresión*	0.11	
CTD	COUNTER	0.43	0.43	
CTU	COUNTER	0.43	0.43	
DDT	DINT		9.1	Basado en el modo TODOS
0 desigualdades		13.0 + (Cmp. longitud * 1.38)		
1 desigualdad		27.6 + (Cmp. longitud * 1.38)		
2 desigualdades		41.2 + (Cmp. longitud * 1.38)		
DEG	REAL	18.5	0.11	
DIV	DINT	10.4	0.10	
	REAL	18.3	0.11	
terminal de datos listo	DINT	2.8	2.8	
EQU	DINT	0.37	0.10	
	REAL	0.37	0.11	
FAL	Refer to the operations within the expression.	10.2 + (*n* * (3.99 + *expresión*)	7.3	*n* es el número de elementos manipulados durante un escán.
FBC	DINT		9.2	
0 desigualdades		13.1 + (Cmp. longitud * 1.38)		Basado en el modo TODOS
1 desigualdad		26.4 + (Cmp. longitud * 1.38)		
2 desigualdades		38.6 + (Cmp. longitud * 1.38)		
FFL	SINT	9.7		
	INT	10.7		
	DINT, REAL	10.1	7.5	
FFU	SINT	10.4 + (longitud * 0.56)		
	INT	11.3 + (longitud * 0.98)		
	DINT, REAL	11.7 + (longitud * 0.56)	7.7	

Table C.2 Tiempos de ejecución de instrucción (Continued)

Instrucción:	Tipo de datos óptimo:	Tiempo verdadero (µs):	Tiempo falso (µs):	Notas:
FLL	SINT	5.8 + (longitud * 0.09)		
	INT	5.6 + (longitud * 0.17)		
	DINT, REAL	6.0 + (longitud * 0.35)	0.16	
For	DINT	16.0 + ((valor terminal/tamaño de paso) * 6.92)	3.0	
FRD	DINT	9.1	0.10	
FSC	Refer to the operations within the expression.	10.7 + (*n* * (3.89 + *expresión*)	7.8	*n* es el número de elementos manipulados durante un escán.
GEQ	DINT	0.37	0.11	
	REAL	0.58	0.11	
GRT	DINT	0.37	0.11	
	REAL	0.59	0.11	
GSV	na	See Table C.3 on page C-8	0.16	
JMP	na	1.6	0.10	
JSR	na	10.62		Sin parámetros
JSR/SBR	SINT	21.0 + (*número de parámetros* * 3.49)		El tiempo es para la pareja JSR/SBR.
	INT	21.0 + (*número de parámetros* * 3.86)		
	DINT	21.0 + (*número de parámetros* * 3.27)		
	REAL	21.0 + (*número de parámetros* * 3.33)		
JSR/RET	SINT	20.4 + (*número de parámetros* * 3.49)		El tiempo es para la pareja JSR/RET.
	INT	20.4 + (*número de parámetros* * 3.86)		
	DINT	20.4 + (*número de parámetros* * 3.27)		
	REAL	20.4 + (*número de parámetros* * 3.33)		
LBL	na	0.16		
LEQ	DINT	0.37	0.11	
	REAL	0.58	0.11	
LES	DINT	0.37	0.10	
	REAL	0.58	0.11	
LFL	SINT	9.7		
	INT	10.7		
	DINT, REAL	10.1	7.5	
LFU	SINT	11.5		
	INT	12.8		
	DINT, REAL	13.7	7.6	
LIM	DINT	0.79	0.11	
	REAL	3.5	0.11	

Table C.2 Tiempos de ejecución de instrucción (Continued)

Instrucción:	Tipo de datos óptimo:	Tiempo verdadero (µs):	Tiempo falso (µs):	Notas:
LN	REAL	189.8	0.10	
LOG	REAL	190.3	0.11	
MCR	na	0.05	0.05	
MEQ	DINT	0.58	0.10	
MOD	DINT	20.1	0.22	
	REAL	63.3	0.20	
MOV	DINT	0.51	0.11	
	REAL	0.88	0.10	
MSG	MESSAGE	93.7	7.0	
MUL	DINT	9.5	0.11	
	REAL	17.7	0.11	
MVM	DINT	2.1	0.16	
NEG	DINT	0.55	0.11	
	REAL	0.93	0.11	
NEQ	DINT	0.37	0.10	
	REAL	0.37	0.11	
NOP	na	0.05	0.06	
NOT	DINT	0.50	0.11	
ONS	BOOL	2.9	2.7	
OR	DINT	0.55	0.11	
OSF	BOOL	3.4	4.0	
OSR	BOOL	3.7	3.5	
OTE	BOOL	0.19	0.19	
OTL	BOOL	0.19	0.05	
OTU	BOOL	0.19	0.05	
PID	PID		18.4	
independiente		349.3		
dependiente de ISA		416.9		
Modo manual		304.6		
Establecer el modo de salida		304.6		
Independiente – modo esclavo		371.0		
RAD	REAL	18.1	0.10	
RES	CONTROL, COUNTER o TIMER	0.32	0.21	
RET (para el lazo FOR)	na	4.55		

Table C.2 Tiempos de ejecución de instrucción (Continued)

Instrucción:	Tipo de datos óptimo:	Tiempo verdadero (µs):	Tiempo falso (µs):	Notas:
RTO	temporizador	0.42	0.31	
SIN	REAL	224.8	0.09	
SQI	DINT	3.6	0.16	
SQL	DINT	6.1	3.9	
SQO	DINT	6.2	3.8	
SQR	DINT	9.9	0.10	
	REAL	37.1	0.11	
SRT	DINT	30.1 + x	4.6	El tiempo “x” varía con la longitud y la naturaleza aleatoria de los números.
	REAL	30.8 + x	4.3	
SSV	na	See Table C.4 on page C-12	0.15	
STD	SINT	104.5 + (longitud * 51.18)	22.0	
	INT	111.9 + (longitud * 51.79)	25.3	
	DINT	112.0 + (longitud * 50.71)	25.3	
	REAL	113.5 + (longitud * 55.08)	25.4	
SUB	DINT	0.55	0.10	
	REAL	10.6	0.11	
TAN	REAL	284.7	0.08	
TND	na	0.00	0.11	
TOD	DINT	14.8	0.10	
TOF	temporizador	0.32	0.42	
TON	temporizador	0.43	0.34	
TRN	DINT	13.0	0.21	
	REAL	21.0	0.21	
UID	na	39.1	2.6	
UIE	na	38.5	2.6	
XIC	BOOL	0.11	0.05	
XIO	BOOL	0.11	0.05	
XOR	DINT	0.56	0.11	
XPY	REAL	274.4	0.10	El tiempo verdadero puede tener un rango de 200 a 400 µs, dependiendo de los valores de los operandos.

Table C.3 Tiempos de ejecución para la instrucción GSV

Objeto:	Atributo:	Tiempo verdadero (µs):
AXIS	AccelerationFeedforwardGain	90.3
AXIS	ActualPosition	160.0
AXIS	ActualVelocity	165.0
AXIS	AverageVelocity	250.2
AXIS	AverageVelocityTimebase	25.6
AXIS	AxisConfigurationState	20.6
AXIS	AxisState	18.7
AXIS	AxisType	20.0
AXIS	C2CConnectionInstance	22.3
AXIS	C2CMapTableInstance	22.6
AXIS	CommandPosition	159.0
AXIS	CommandVelocity	164.0
AXIS	ConversionConstant	22.7
AXIS	DampingFactor	20.3
AXIS	DriveFaultAction	21.0
AXIS	EffectiveInertia	22.4
AXIS	EncoderLossFaultAction	19.6
AXIS	EncoderNoiseFaultAction	20.7
AXIS	FrictionCompensation	83.7
AXIS	GroupInstance	22.5
AXIS	HomeMode	20.3
AXIS	HomePosition	159.9
AXIS	HomeReturnSpeed	34.2
AXIS	HomeSequenceType	19.6
AXIS	HomeSpeed	33.9
AXIS	Instance	18.6
AXIS	MapTableInstance	22.6
AXIS	MaximumAcceleration	62.4
AXIS	MaximumDeceleration	62.7
AXIS	MaximumNegativeTravel	160.5
AXIS	MaximumPositiveTravel	160.2
AXIS	MaximumSpeed	61.2
AXIS	ModuleChannel	21.0
AXIS	MotionConfigurationBits	21.3
AXIS	MotionFaultBits	22.1

Table C.3 Tiempos de ejecución para la instrucción GSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (μs):
AXIS	MotionStatusBits	22.4
AXIS	MotorEncoderTestIncrement	159.6
AXIS	OutputFilterBandwidth	22.5
AXIS	OutputLimit	82.3
AXIS	OutputOffset	83.2
AXIS	OutputScaling	200.8
AXIS	PositionError	110.9
AXIS	PositionErrorFaultAction	21.3
AXIS	PositionErrorTolerance	198.6
AXIS	PositionIntegralGain	191.4
AXIS	PositionIntegratorError	112.4
AXIS	PositionLockTolerance	159.8
AXIS	PositionProportionalGain	119.2
AXIS	PositionServoBandwidth	21.8
AXIS	PositionUnwind	21.9
AXIS	ProgrammedStopMode	20.3
AXIS	RegistrationPosition	159.7
AXIS	ServoConfigurationBits	22.9
AXIS	ServoConfigurationUpdateB	21.9
AXIS	ServoEventBits	21.6
AXIS	ServoFaultBits	21.7
AXIS	ServoOutputLevel	108.2
AXIS	ServoStatusBits	22.3
AXIS	ServoStatusUpdateBits	21.4
AXIS	SoftOvertravelFaultAction	21.0
AXIS	StartActualPosition	160.0
AXIS	StartCommandPosition	158.7
AXIS	StrobeActualPosition	160.0
AXIS	StrobeCommandPosition	159.0
AXIS	TestDirectionForward	20.3
AXIS	TestStatus	19.8
AXIS	TuneAcceleration	186.8
AXIS	TuneAccelerationTime	91.8
AXIS	TuneDeceleration	187.0
AXIS	TuneDecelerationTime	92.5
AXIS	TuneRiseTime	93.1

Table C.3 Tiempos de ejecución para la instrucción GSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (µs):
AXIS	TuneSpeedScaling	200.5
AXIS	TuneStatus	20.2
AXIS	TuneVelocityBandwidth	20.9
AXIS	TuningConfigurationBits	22.1
AXIS	TuningSpeed	165.0
AXIS	TuningTravelLimit	159.9
AXIS	VelocityCommand	111.4
AXIS	VelocityError	111.4
AXIS	VelocityFeedback	112.7
AXIS	VelocityFeedforwardGain	90.6
AXIS	VelocityIntegralGain	191.3
AXIS	VelocityIntegratorError	111.6
AXIS	VelocityProportionalGain	119.3
AXIS	WatchPosition	159.7
CONTROLLER	TimeSlice	16.9
CONTROLLERDEVICE	DeviceName	55.2
CONTROLLERDEVICE	ProductCode	15.4
CONTROLLERDEVICE	ProductRev	15.4
CONTROLLERDEVICE	SerialNumber	16.3
CONTROLLERDEVICE	Status	15.4
CONTROLLERDEVICE	Tipo	15.4
CONTROLLERDEVICE	Vendor	15.4
CST	CurrentStatus	14.4
CST	CurrentValue	28.2
DF1	ACTTimeout	16.5
DF1	DiagnosticCounters	67.2
DF1	DuplicationDetect	14.9
DF1	EmbeddedResponseEnable	14.9
DF1	ENQTransmitLimit	14.9
DF1	EOTSuppression	14.9
DF1	ErrorDetection	14.9
DF1	MasterMessageTransmit	14.9
DF1	NAKReceiveLimit	14.9
DF1	NormalPollGroupSize	15.7
DF1	PollingMode	14.9
DF1	ReplyMessageWait	16.5

Table C.3 Tiempos de ejecución para la instrucción GSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (μs):
DF1	StationAddress	15.7
DF1	SlavePollTimeout	16.5
DF1	TransmitRetries	14.9
FAULTLOG	MajorEvents	16.7
FAULTLOG	MinorEvents	16.7
FAULTLOG	MajorFaultBits	17.5
FAULTLOG	MajorFaultBits	17.5
MESSAGE	ConnectionPath	53.1
MESSAGE	ConnectionRate	17.4
MESSAGE	MessageType	15.7
MESSAGE	Puerto	15.7
MESSAGE	TimeoutMultiplier	15.7
MESSAGE	UnconnectedTimeout	17.4
MODULE	EntryStatus	16.7
MODULE	FaultCode	16.7
MODULE	FaultInfo	17.6
MODULE	ForceStatus	144.5
MODULE	Instance	17.8
MODULE	Mode	16.7
MODULE	LEDStatus	17.2
MOTIONGROUP	Instance	24.2
PROGRAM	DisableFlag	16.4
PROGRAM	Instance	17.8
PROGRAM	LastScanTime	17.6
PROGRAM	MajorFaultRecord	59.0
PROGRAM	MaxScanTime	17.6
PROGRAM	MinorFaultRecord	59.0
PROGRAM	SFCRestart	16.7
ROUTINE	Instance	16.8
SERIALPORT	BaudRate	16.7
SERIALPORT	DataBits	15.0
SERIALPORT	Parity	15.0
SERIALPORT	RTSOffDelay	15.8
SERIALPORT	RTSSendDelay	15.8
SERIALPORT	StopBits	15.0
TASK	Instance	17.7

Table C.3 Tiempos de ejecución para la instrucción GSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (µs):
TASK	LastScanTime	17.5
TASK	MaxInterval	21.6
TASK	MaxScanTime	17.4
TASK	MinxInterval	21.6
TASK	Priority7	16.6
TASK	Rate	17.4
TASK	StartTime	21.6
TASK	Watchdog	17.4
WALLCLOCKTIME	CSTOffset	21.2
WALLCLOCKTIME	CurrentValue	37.6
WALLCLOCKTIME	DateTime	59.8

Table C.4 Tiempo de ejecución para la instrucción SSV

Objeto:	Atributo:	Tiempo verdadero (µs):
AXIS	AccelerationFeedforwardGain	105.4
AXIS	AverageVelocityTimebase	168.1
AXIS	AxisType	39.2
AXIS	ConversionConstant	126.9
AXIS	DampingFactor	129.8
AXIS	DriveFaultAction	40.3
AXIS	EncoderLossFaultAction	40.2
AXIS	EncoderNoiseFaultAction	40.0
AXIS	FrictionCompensation	140.1
AXIS	HomeMode	38.2
AXIS	HomePosition	223.1
AXIS	HomeReturnSpeed	511.7
AXIS	HomeSequenceType	37.6
AXIS	HomeSpeed	511.3
AXIS	MaximumAcceleration	511.5
AXIS	MaximumDeceleration	511.8
AXIS	MaximumNegativeTravel	224.7
AXIS	MaximumPositiveTravel	224.7
AXIS	MaximumSpeed	511.7
AXIS	MotionConfigurationBits	38.8

Table C.4 Tiempo de ejecución para la instrucción SSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (μs):
AXIS	MotorEncoderTestIncrement	207.0
AXIS	OutputFilterBandwidth	45.7
AXIS	OutputLimit	139.4
AXIS	OutputOffset	140.2
AXIS	OutputScaling	383.1
AXIS	PositionErrorFaultAction	39.2
AXIS	PositionErrorTolerance	202.7
AXIS	PositionIntegralGain	218.0
AXIS	PositionLockTolerance	258.1
AXIS	PositionProportionalGain	153.1
AXIS	PositionServoBandwidth	44.1
AXIS	PositionUnwind	128.0
AXIS	ProgrammedStopMode	38.6
AXIS	ServoConfigurationBits	42.2
AXIS	SoftOvertravelFaultAction	40.3
AXIS	TuningConfigurationBits	38.9
AXIS	TuningSpeed	580.6
AXIS	TuningTravelLimit	207.2
AXIS	VelocityFeedforwardGain	106.1
AXIS	VelocityIntegralGain	218.0
AXIS	VelocityProportionalGain	152.1
CONTROLLER	TimeSlice	35.9
DF1	PendingACKTimeout	109.4
DF1	PendingDuplicateDetection	108.3
DF1	PendingEmbeddedResponseEnable	108.7
DF1	PendingENQTransmitLimit	108.3
DF1	PendingEOTSuppression	108.2
DF1	PendingErrorDetection	108.9
DF1	PendingNormalPollGroupSize	108.9
DF1	PendingMasterMessageTransmit	108.7
DF1	PendingNAKReceiveLimit	108.3
DF1	PendingPollingMode	108.7
DF1	PendingReplyMessageWait	109.4
DF1	PendingStationAddress	109.1
DF1	PendingSlavePollTimeout	109.4
DF1	PendingTransmitRetries	108.3

Table C.4 Tiempo de ejecución para la instrucción SSV (Continued)

Objeto:	Atributo:	Tiempo verdadero (µs):
FAULTLOG	MajorEvents	17.0
FAULTLOG	MinorEvents	17.0
FAULTLOG	MajorFaultBits	17.4
FAULTLOG	MajorFaultBits	17.4
MESSAGE	ConnectionPath	36.9
MESSAGE	ConnectionRate	32.3
MESSAGE	MessageType	69.9
MESSAGE	Port	31.6
MESSAGE	TimeoutMultiplier	31.7
MESSAGE	UnconnectedTimeout	32.3
MODULE	Mode	28.4
PROGRAM	DisableFlag	37.3
PROGRAM	LastScanTime	17.5
PROGRAM	MajorFaultRecord	48.8
PROGRAM	MaxScanTime	17.5
PROGRAM	MinorFaultRecord	48.8
PROGRAM	SFCRestart	33.0
SERIALPORT	PendingBaudRate	108.5
SERIALPORT	PendingDataBits	106.7
SERIALPORT	PendingParity	106.6
SERIALPORT	PendingRTSOffDelay	106.8
SERIALPORT	PendingRTSSendDelay	107.4
SERIALPORT	PendingStopBits	106.7
TASK	LastScanTime	17.7
TASK	MaxInterval	24.6
TASK	MaxScanTime	17.7
TASK	MinxInterval	24.6
TASK	StartTime	24.6
TASK	Watchdog	33.8
WALLCLOCKTIME	CSTOffset	2149.4
WALLCLOCKTIME	CurrentValue	70.5
WALLCLOCKTIME	DateTime	120.0

Table C.5 Tiempos de índice de arreglo

Para el tag en el subíndice de un:	Añada (µs):
arreglo de una dimensión	1.5
un arreglo de dos dimensiones	25.5
un arreglo de tres dimensiones	30.3

Notas:

Uso de memoria

Introducción

Use este apéndice para calcular la memoria usada para las conversiones de datos, instrucciones e índices de arreglo. Los valores son para un controlador Logix5550 (No. de cat. 1756-L1) y el software RSLogix 5000, versión 2.25.

▶ Cuando se descarga el proyecto, el controlador optimiza el uso de memoria, lo cual puede reducir el uso de memoria calculado en hasta 10 % de los valores presentados en este apéndice.

Para calcular la memoria para:	Vea la página:
conversiones de datos	D-2
instrucciones de lógica de escalera	D-3
índices de arreglos	D-7

Conversiones de datos

Si usted usa un tipo de datos que no es un tipo de datos óptimo o si combina los tipos de datos, tiene que añadir memoria para la conversión de datos. Consulte “Conversiones de datos” en la página A-6.

Requisitos de memoria de extensión con signo

Use la siguiente tabla para determinar la memoria requerida para convertir los datos usando una extensión con signo.

Para convertir un:	De un:	A un:	Añada (bytes):
Operando de origen	SINT	DINT	48
		REAL	96
	INT	DINT	60
		REAL	108
	DINT	REAL	52
Operando de destino	DINT	SINT	52
		INT	52
		REAL	76
	REAL	SINT	48
		INT	48
		DINT	48

Requisitos de memoria de relleno con ceros

Use la siguiente tabla para determinar la memoria requerida para convertir los datos usando un relleno con ceros.

Para convertir un:	De un:	A un:	Añada (bytes):
Operando de origen	SINT	DINT	0
	INT	DINT	8
Operando de destino	DINT	SINT	52
		INT	52

Ejemplos:

Instrucción:	Tipo de datos:	Memoria (bytes):
ADD	REAL	44 bytes para la instrucción ADD
Origen A	DINT	52 bytes para la conversión de DINT a REAL usando la extensión con signo
Origen B	REAL	
Destino	REAL	
		total de 96 bytes
MUL	REAL	44 bytes para la instrucción MUL
Origen A	REAL	
Origen B	DINT	52 bytes para la conversión de DINT a REAL usando la extensión con signo
Destino	DINT	48 bytes para la conversión de REAL a DINT usando la extensión con signo
		total de 144 bytes

Instrucciones

La siguiente tabla muestra el uso de memoria de las instrucciones de lógica de escalera. Para las instrucciones que usan una expresión (CMP, CPT, FAL, FSC), añada memoria para cada operación dentro de la expresión. Para cada operación, use el valor para la instrucción correspondiente.

Ejemplo:

Instrucción:	Tipo de datos:	Memoria:
CMP (*tag_a * tag_b / 100)*		tiempo de procesamiento interno de 76 bytes
* (MUL)	REAL	44 bytes para la operación MUL
/ (DIV)	REAL	44 bytes para la operación DIV
		total de 164 bytes

Categoría:	Mnemónico:	Título:	Memoria para DINT (bytes):	Memoria para REAL (bytes):
Renglón y bifurcación		Rung	16	
		Branch Start	4	
		Next Branch	4	
		Branch End	4	

Categoría:	Mnemónico:	Título:	Memoria para DINT (bytes):	Memoria para REAL (bytes):
bit	XIC	Examine If Open	4	
	XIO	Examine If Open	4	
	OTE	Output Energize	4	
	OTL	Output Latch	4	
	OTU	Output Unlatch	4	
	ONS	One Shot	36	
	OSR	One Shot Rising	44	
	OSF	One Shot Falling	44	
Temporizador y contador	TON	Timer On Delay	8	
	TOF	Timer Off Delay	8	
	RTO	Retentive Timer On	8	
	CTU	Count Up	8	
	CTD	Count Down	8	
	RES	Reset	8	
Entrada y salida	MSG	Message	36	
	GSV	Get System Value	76	76
	SSV	Set System Value	76	76
Comparación	CMP	Comparación	76 + expresión	76 + expresión
	EQU	Equal to	20	20
	GEQ	Greater Than or Equal to	20	36
	GRT	mayor que	20	36
	LEQ	Less Than or Equal to	20	36
	LES	Less Than	20	36
	LIM	Limit	52	44
	MEQ	Mask Equal to	32	
	NEQ	Not Equal to	20	20
Cálculo	CPT	Cálculo	76 + expresión	96 + expresión
	ADD	Add	28	44
	SUB	Subtract	28	44
	MUL	Multiply	44	44
	DIV	Divide	44	44
	MOD	Módulo	44	44
	SQR	Square Root	40	40
	NEG	Negate	28	56
	ABS	Absolute Value	32	56

Categoría:	Mnemónico:	Título:	Memoria para DINT (bytes):	Memoria para REAL (bytes):
Transferencia/ lógica	MOV	Move	24	48
	MVM	Masked Move	80	
	BTD	Bit Field Distribute	52	
	CLR	Clear	20	20
	AND	Bitwise AND	28	
	OR	Bitwise OR	28	
	XOR	Bitwise Exclusive OR	28	
	NOT	NOT bit a bit	28	
Arreglo (archivo)/ Misceláneos	FAL	File Arithmetic and Logic	92 + expresión	116 + expresión
	FSC	File Search and Compare	148 + expresión	152 + expresión
	COP	Copy	64	64
	FLL	Fill	60	60
	AVE	Average	164	116
	SRT	Sort	108	108
	STD	Standard Deviation	104	104
Desplazamiento de archivo	BSL	Bit Shift Left	52	
	BSR	Bit Shift Right	52	
	FFL	FIFO Load	64	64
	FFU	FIFO Unload	64	64
	LFL	LIFO Load	64	64
	LFU	LIFO Unload	64	64
Secuenciador	SQI	Sequencer Input	48	
	SQL	Sequencer Load	48	
	SQO	Sequencer Output	44	
Control del programa	JMP	Jump to Label	24	
	LBL	Label	12	
	JSR	Jump to Subroutine	56	
	SBR	Subrutina	40	
	RET	Return	40	
	AFI	Always False	4	
	MCR	Master Control Reset	4	
	NOP	No Operation	4	
	TND	Temporary End	12	
	UID	User Interrupt Disabled	28	
	UIE	User Interrupt Enabled	28	
For/Interrumpir	For	For	64	
	BRK	Break	36	

Categoría:	Mnemónico:	Título:	Memoria para DINT (bytes):	Memoria para REAL (bytes):
Especiales	FBC	File Bit Comparison	72	
	DDT	Diagnostic Detect	72	
	DTR	Data Transitional	40	
	PID	Proportional, Integral, Derivative	228	92
Trigonométricas	SIN	Sine	140	48
	COS	Cosine	140	48
	TAN	Tangent	140	48
	ASN	Arc Sine	140	48
	ACS	Arc Cosine	140	48
	ATN	Arc Tangent	140	48
Matemáticas avanzadas	LN	Natural Log	140	48
	LOG	Log Base 10	140	48
	XPY	X to the Power of Y	144	52
Conversión matemática	DEG	Radians to Degrees	144	52
	RAD	Degrees to Radians	144	52
	TOD	Integer to BCD	40	
	FRD	BCD to Integer	40	
	TRN	Truncate	40	92

Indices de arreglos

El uso de memoria para una instrucción se incrementa por cada parámetro que hace referencia a un arreglo. Tome en cuenta lo siguiente:

- Cuando un arreglo usa los valores inmediatos para el índice, no hay memoria adicional añadida.

  Por ejemplo:

Ejemplo:	Memoria adicional usada:
myTag[0]	ninguna
myTag[0,0,0]	ninguna

- Cuando un arreglo usa un tag (tipo DINT) como índice, se usa la memoria adicional según el número de dimensiones en el arreglo.

  Por ejemplo:

Ejemplo:	Memoria adicional usada:
myTag[offset]	84 bytes
myTag[0,offset,0]	152 bytes

- El controlador usa un índice de 32 bits para todos los arreglos. Si se usa un tag que no sea DINT para hacer referencia a una posición en un arreglo, se añade el uso de memoria de conversión de datos (vea la página D-2) así como el uso de memoria para usar un tag como índice.

  Por ejemplo:

TagA	DINT
TagB	DINT
TagC	INT
TableA	DINT[10]
TableB	DINT[2,4,6]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV Origen Destino	 TagA TagB	24 bytes para la instrucción MOV no se añade nada más total de 24 bytes
Instrucción MOV Origen Destino	 TagA TableA0	24 bytes para la instrucción MOV no se añade nada más total de 24 bytes
Instrucción MOV Origen Destino	 TagA TableA[TagB]	24 bytes para la instrucción MOV 84 bytes para el índice de tag DINT (TableA) total de 108 bytes
Instrucción MOV Origen Destino	 TagA TableA[TagC]	24 bytes para la instrucción MOV 60 bytes para la conversión de INT a DINT (TagC) 84 bytes para el índice de tag DINT (TableA) total de 168 bytes
Instrucción MOV Origen Destino	 TagA TableB[0,0,0]	24 bytes para la instrucción MOV no se añade nada más total de 24 bytes
Instrucción MOV Origen Destino	 TagA TableB[TagB,0,0]	24 bytes para la instrucción MOV 152 bytes para el índice de tag DINT (TableB) total de 176 bytes
Instrucción MOV Origen Destino	 TagA TableB[TagC,0,0]	24 bytes para la instrucción MOV 60 bytes para la conversión de INT a DINT (TagC) 152 bytes para el índice de tag DINT (TableB) total de 236 bytes

- El cambiar el tipo de datos de origen y destino en los ejemplos anteriores aumenta el uso de memoria de la instrucción.

  Por ejemplo:

TagA	DINT
TagC	INT
TagD	SINT
TableC	INT[10]
TableD	SINT[2,4,6]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV Origen Destino	 TagA TableC[TagC]	24 bytes para la instrucción MOV 52 bytes para la conversión de DINT a INT (TagA en TableC) 60 bytes para la conversión de INT a DINT (TagC) 84 bytes para el índice de tag DINT (TableC) total de 220 bytes
Instrucción MOV Origen Destino	 TagD TableC[TagC]	24 bytes para la instrucción MOV 92 bytes para la conversión de SINT a INT (TagD en TableC) 60 bytes para la conversión de INT a DINT (TagC) 84 bytes para el índice de tag DINT (TableC) total de 260 bytes
Instrucción MOV Origen Destino	 TagA TableD[TagC,0,0}	24 bytes para la instrucción MOV 52 bytes para la conversión de DINT a SINT (TagA en TableD) 60 bytes para la conversión de INT a DINT (TagC) 152 bytes para el índice de tag DINT (TableD) total de 288 bytes
Instrucción MOV Origen Destino	 TagC TableD[TagC,0,0]	24 bytes para la instrucción MOV 104 bytes para la conversión de INT a SINT (TagC en TableC) 60 bytes para la conversión de INT a DINT (TagC) 152 bytes para el índice de tag DINT (TableD) total de 340 bytes

- Si se usa una expresión en el índice, se usa memoria adicional según los operadores.

  Por ejemplo:

TagA	DINT
TagB	DINT
TagF	DINT
TableA	DINT[10]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV		24 bytes para la instrucción MOV
Origen	TagA	84 bytes para el índice de tag DINT (TableA)
Destino	TableA[TagB + TagF]	28 bytes para ADD
		total de 136 bytes

- Si usa múltiples arreglos en una instrucción, añada el uso de memoria apropiado para cada referencia de arreglo.

  Por ejemplo:

TagA	DINT
TagB	DINT
TableA	DINT[10]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV		24 bytes para la instrucción MOV
Origen	TableA[TagA]	84 bytes para el índice de tag DINT (TableA)
Destino	TableA[TagB]	84 bytes para el segundo índice de tag DINT (TableA)
		total de 192 bytes

- El peor de casos ocurre cuando hay múltiples referencias a arreglos multidimensionales que no son del tipo DINT con índices que no son del tipo DINT en dichos arreglos.

  Por ejemplo:

TagC	INT
TagD	SINT
TagE	INT
TableD	SINT[2,4,6]
TableE	INT[2,4,6]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV		24 bytes para la instrucción MOV
Origen	TableD[TagC, TagD, TagE]	60 bytes para la conversión de INT a DINT (TagC)
Destino	TableE[TagC, tagD, TagE]	48 bytes para la conversión de SINT a DINT (TagD)
		60 bytes para la conversión de INT a DINT (TagE)
		152 bytes para el índice de tag DINT (TableD)
		60 bytes para la conversión de INT a DINT (TagC)
		48 bytes para la conversión de SINT a DINT (TagD)
		60 bytes para la conversión de INT a DINT (TagE)
		152 bytes para el índice de tag DINT (TableD)
		92 bytes para la conversión SINT a INT (TableD en TableE)
		total de 756 bytes

- Puede optimizar esta instrucción haciendo que los dos arreglos sean del tipo DINT con índices de tag DINT. Por ejemplo:

TagA	DINT
TagB	DINT
TagF	DINT
TableB	DINT[2,4,6]
TableF	DINT[2,4,6]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV		24 bytes para la instrucción MOV
Origen	TableB[TagA, TagB, TagF]	152 bytes para el índice de tag DINT (TableB)
Destino	TableF[TagA, TagB, TagF}	152 bytes para el índice de tag DINT (TableD)
		total de 328 bytes

- Puede optimizar aún más esta instrucción cambiando todos los índices de arreglos a valores inmediatos. Por ejemplo:

TagA	DINT
TagB	DINT
TagF	DINT
TableB	DINT[2,4,6]
TableF	DINT[2,4,6]

Ejemplo:		**Memoria adicional usada:**
Instrucción MOV		24 bytes para la instrucción MOV
Origen	TableB[0,1,2]	no se añade nada más
Destino	TableF[3,4,5]	total de 24 bytes

A

activación de salida 1-6
ajuste 12-24
alarmas 12-25
arco coseno 13-10
arco seno 13-8
arco tangente 13-12
aritmética y lógica de archivos 7-6
arreglo B-6
asignación de memoria B-6
atributos
  conversión de tipos de datos A-6
  palabras clave A-4
  palabras clave de estado aritmético A-1
  tipos de datos A-5
atributos comunes
  conversión de tipos de datos A-6
  palabras clave A-4
  palabras clave de estado aritmético A-1
  tipos de datos A-5

B

banda muerta 12-34
bit a bit
  AND 6-9
  NOT 6-15
  OR 6-11
  XOR 6-13
bit dentro de arreglos B-4
borrar 6-8
buscar y comparar archivos 7-17

C

cálculo 5-2
cambiar signo 5-18
carga de secuenciador 9-11
carga FIFO 8-8
carga LIFO 8-20
clasificar 7-38
códigos de error
  instrucción MSG 3-7
combinación de tipos de datos A-6
comparación 4-2
comparación de bits de archivo 12-2
conceptos de arreglo
  asignación de memoria B-6
  especificación de bit B-4
  índice B-3
  uso de memoria D-7
configuración 3-13
  instrucción MSG 3-13
  instrucción PID 12-24
contador 2-1
conteo progresivo 2-11
conteo regresivo 2-14
control maestro de reset 10-11
conversión de BCD 15-6
conversión de tipos de datos A-6
conversión en BCD 15-4
conversiones de datos D-2
copiar 7-27
coseno 13-4

D

datos de transición 12-16
definir valor del sistema 3-27
descarga FIFO 8-14
descarga LIFO 8-26
desenclavamiento de salida 1-8
desplazamiento a la derecha 8-5
desplazamiento a la izquierda 8-2
desviación estándar 7-41
detección de diagnóstico 12-9
distribución de campos de bits 6-5
dividir 5-12

E

enclavamiento de salida 1-7
escala 12-26
estructura CONTROL 7-6, 7-17, 7-34, 7-38, 7-41, 8-2, 8-5, 8-8, 8-14, 8-20, 8-26, 9-2, 9-6, 9-11, 12-3, 12-10
estructura COUNTER 2-11, 2-14
estructura MESSAGE 3-2
estructura TIMER 2-2, 2-5, 2-8

estructuras
CONTROL 7-6, 7-17, 7-34, 7-38, 7-41, 8-2, 8-5, 8-8, 8-14, 8-20, 8-26, 9-2, 9-6, 9-11, 12-3, 12-10
COUNTER 2-11, 2-14
instrucción RES 2-18
MSG 3-2
PID 12-20
TIMER 2-2, 2-5, 2-8
etiqueta 10-2
examina si abierto 1-4
examinar si cerrado 1-2
exponencial 14-6
expresiones
formato 4-4, 5-4, 7-15, 7-25
operadores válidos 4-4, 5-4, 7-15, 7-25
orden de operación 4-5, 5-5, 7-16, 7-26

F

fin temporal 10-10

G

grados en radianes 15-3

I

igual a 4-6
índice B-3
instrucción ABS 5-19
instrucción ACS 13-10
instrucción ADD 5-6
instrucción AFI 10-15
instrucción AND 6-9
instrucción ASN 13-8
instrucción ATRN 13-12
instrucción AVE 7-34
instrucción BRK 11-5
instrucción BSL 8-2
instrucción BSR 8-5
instrucción BTD 6-5
instrucción CLR 6-8
instrucción CMP 4-2
instrucción COP 7-27
instrucción COS 13-4
instrucción CPT 5-2
instrucción CTD 2-14
instrucción CTU 2-11
instrucción DDT
modo buscar 12-11
operandos 12-9
instrucción de secuenciador SQL 9-11
instrucción DEG 15-2
instrucción DIV 5-12
instrucción DTR 12-16
instrucción EQU 4-6
instrucción FAL
modo de operación 7-1
operandos 7-6
instrucción FBC
modo buscar 12-4
operandos 12-2
instrucción FFL 8-8
instrucción FFU 8-14
instrucción FLL 7-31
instrucción FOR 11-2
instrucción FRD 15-6
instrucción FSC 7-17
instrucción GEQ 4-8
instrucción GRT 4-10
instrucción GSV
objetos 3-29
instrucción JMP 10-2
instrucción JSR 10-4
instrucción LBL 10-2
instrucción LEQ 4-12
instrucción LFL 8-20
instrucción LIM 4-16
instrucción LN 14-2
instrucción LOG 14-4
instrucción MCR 10-11
instrucción MEQ 4-19
instrucción MOD 5-14
instrucción MOV 6-2
instrucción MSG 3-13
códigos de error 3-7
conexión de caché 3-26
estructura 3-2
método de comunicación 3-25
operandos 3-2
instrucción MUL 5-10
instrucción MVM 6-3
instrucción NEG 5-18
instrucción NEQ 4-22
instrucción NOP 10-16
instrucción NOT 6-15
instrucción ONS 1-9
instrucción OR 6-11

instrucción OSF 1-14
instrucción OSR 1-11
instrucción OTE 1-6
instrucción OTL 1-7
instrucción OTU 1-8
instrucción PID
  ajuste 12-24
  alarmas 12-25
  banda muerta 12-34
  configuración 12-24
  escala 12-26
  estructura 12-20
  operandos 12-19
  polarización de salida 12-35
  prealimentación 12-35
instrucción RAD 15-3
instrucción RES 2-18
instrucción RET 10-4, 11-6
instrucción RTO 2-8
instrucción SBR 10-4
instrucción SIN 13-2
instrucción SQI
  entrada de secuenciador 9-2
instrucción SQO 9-6
instrucción SQR 5-16
instrucción SRT 7-38
instrucción SSV
  objetos 3-29
  operandos 3-27
instrucción SUB 5-8
instrucción TAN 13-6
instrucción TND 10-10
instrucción TOD 15-4
instrucción TOF 2-5
instrucción TRN 15-8
instrucción UID 10-13
instrucción UIE 10-14
instrucción XIC 1-2
instrucción XIO 1-4
instrucción XOR 6-13
instrucción XPY 14-6
instrucciones 2-1
  arreglo
  atributos comunes A-1
  bit 1-1
  cálculo 5-1
  comparación 4-1
  control de programa 10-1
  conversión 15-1
  conversión matemática 15-1
  desplazamiento 8-1
  entrada/salida 3-1
  especiales 12-1
  FOR/interrupción 11-1
  lógica 6-1
  matemáticas avanzadas 14-1
  secuenciador 9-1
  temporizador 2-1
  tiempo de ejecución C-1
  transferencia 6-1
  trigonométricas 13-1
  uso de memoria D-3
instrucciones de archivo. Vea las instrucciones de arreglo
instrucciones de arreglo
  archivo/misceláneos 7-1
  AVE 7-34
  BSL 8-2
  BSR 8-5
  conceptos de arreglo B-1
  COP 7-27
  DDT 12-9
  desplazamiento 8-1
  FAL 7-6
  FBC 12-2
  FFL 8-8
  FFU 8-14
  FLL 7-31
  FSC 7-17
  LFL 8-20
  LFU 8-26
  modo de operación 7-1
  secuenciador 9-1
  SQI 9-2
  SQL 9-11
  SQO 9-6
  SRT 7-38
  STD 7-41
instrucciones de bit
  introducción 1-1
  ONS 1-9
  OSF 1-14
  OSR 1-11
  OSR 1-11
  OTE 1-6
  OTL 1-7
  OTU 1-8
  XIC 1-2
  XIO 1-4

instrucciones de cálculo
ABS 5-19
ADD 5-6
CPT 5-2
DIV 5-12
formato de expresión 5-4, 7-15
introducción 5-1
MOD 5-14
MUL 5-10
NEG 5-18
operadores válidos 5-4, 7-15
orden de operación 5-5, 7-16
SQR 5-16
SUB 5-8
instrucciones de comparación
CMP 4-2
EQU 4-6
formato de expresión 4-4, 7-25
GEQ 4-8
GRT 4-10
introducción 4-1
LEQ 4-12
LES, instrucción LES 4-14
LIM 4-16
MEQ 4-19
NEQ 4-22
orden de operación 4-5, 7-26
instrucciones de contador
introducción 2-1
instrucciones de control de programa
introducción 10-1
instrucciones de control maestro de reset
MCR 10-11
instrucciones de conversión 15-4
DEG 15-2
FRD 15-6
introducción 15-1
RAD 15-3
TRN 15-8
instrucciones de conversión matemática
introducción 15-1
TRN 15-8
instrucciones de conversiones matemáticas
DEG 15-2
FRD 15-6
RAD 15-3
TOD 15-4
instrucciones de desplazamiento
BSL 8-2
BSR 8-5
FFL 8-8
FFU 8-14
introducción 8-1
LFL 8-20
LFU 8-26
instrucciones de entrada/salida
GSV 3-27
introducción 3-1
MSG 3-2
SSV 3-27
instrucciones de lógica
introducción 6-1
instrucciones de mover
BTD 6-5
CLR 6-8
MOV 6-2
MVM 6-3
instrucciones de secuenciador
introducción 9-1
SQI 9-2
SQL 9-11
SQO 9-6
instrucciones de temporizador
introducción 2-1
TON 2-2
instrucciones de transferencia
introducción 6-1
instrucciones del arreglo
RES 2-18
instrucciones del contador
CTD 2-14
CTU 2-11
RES 2-18
instrucciones del temporizador
RES 2-18
RTO 2-8
TOF 2-5
instrucciones especiales
DDT 12-9
DTR 12-16
FBC 12-2
introducción 12-1
PID 12-19
instrucciones GSV
operandos 3-27

instrucciones lógicas
AND 6-9
NOT 6-15
OR 6-11
XOR 6-13
instrucciones matemáticas avanzadas
introducción 14-1
LN 14-2
LOG 14-4
XPY 14-6
instrucciones para el control del programa 10-4
AFI 10-15
JMP 10-2
JSR 10-4
LBL 10-2
NOP 10-16
SBR 10-4
TND 10-10
UID 10-13
UIE 10-14
instrucciones STD 7-41
instrucciones TON 2-2
instrucciones trigonométricas
ACS 13-10
ASN 13-8
ATN 13-12
COS 13-4
introducción 13-1
SIN 13-2
TAN 13-6
interrupción 11-5
interrupción de usuario activada 10-14
interrupción de usuario desactivada 10-13

L

límite 4-16
llenar archivo 7-31
logaritmo de base 10 14-4
logaritmo natural 14-2

M

máscara igual a 4-19
máscaras 4-19, 6-3, 12-16
mayor que 4-10
mayor que o igual a 4-8
menor que o igual a 4-12
menos que 4-14
mensaje 3-2
modo buscar 12-4, 12-11
modo de operación 7-1
modo incremental 7-4
modo numérico 7-3
modo todos 7-2
módulo 5-14
mover con máscara 6-3
multiplicar 5-10

N

no igual a 4-22

O

O exclusivo 6-13
objeto AXIS 3-30
objeto CONTROLLER 3-37
objeto CONTROLLERDEVICE 3-37
objeto CST 3-39
objeto DF1 3-40
objeto FAULTLOG 3-43
objeto MESSAGE 3-43
objeto MODULE 3-45
objeto MOTIONGROUP 3-46
objeto PROGRAM 3-47
objeto ROUTINE 3-47
objeto SERIALPORT 3-48
objeto TASK 3-49
objeto WALLCLOCKTIME 3-50
objetos
AXIS 3-30
CONTROLLER 3-37
CONTROLLERDEVICE 3-37
CST 3-39
DF1 3-40
FAULTLOG 3-43
instrucción GSV/SSV 3-29
MESSAGE 3-43
MODULE 3-45
MOTIONGROUP 3-46
PROGRAM 3-47
ROUTINE 3-47
SERIALPORT 3-48
TASK 3-49
WALLCLOCKTIME 3-50
obtener valor del sistema 3-27
operadores 5-4, 7-15
instrucciones de comparación operadores válidos 4-4, 7-25
orden de operación 4-5, 5-5, 7-16, 7-26

P
palabras clave
estado aritmético A-1
otras A-4
palabras clave de estado aritmético A-1
polarización de salida 12-35
prealimentación 12-35
promedio 7-34
proporcional, integral y derivada 12-19

R
radianes en grados 15-2
raíz cuadrada 5-16
relleno de cero A-6
restablecimiento 2-18
restar 5-8
RET 10-4
retorno 10-4

S
salida de secuenciador 9-6
saltar a subrutina 10-4
salto 10-2
seno 13-2
siempre falso 10-15
sin operación 10-16
subrutina 10-4
sumar 5-6

T
tangente 13-6
temporizador a la conexión 2-2
temporizador a la desconexión 2-5
temporizador retentivo a la conexión 2-8
tiempo
ejecución C-1
tiempo de ejecución C-1
tipos de datos A-5
truncar un valor 15-8

U
un frente ascendente 1-11
un frente descendente 1-14
un impulso 1-9
uso de memoria C-1, D-1

V
valor absoluto 5-19

X
X a la potencia de Y 14-6

Notas:

Más encontrará en www.rockwellautomation.com

En cualquier lugar en el que nos necesite, Rockwell Automation reúne las marcas líder en automatización industrial, incluyendo los controles Allen-Bradley, los productos de transmisión de potencia eléctrica Reliance Electric, los componentes de transmisión de potencia mecánica Dodge y los programas de Rockwell Software. La manera única y flexible en la que Rockwell Automation ayuda a sus clientes a lograr una ventaja competitiva está respaldada por miles de socios, distribuidores e integradores de sistemas autorizados en todo el mundo.

Sede central: 1201 South Second Street, Milwaukee, WI 53204, USA, Tel: (1) 414-382-2000, Fax: (1) 414-382-4444
Sede central europea: 46, avenue Hermann Debroux, 1160 Bruselas, Bélgica, Tel: (32) 2 663 06 00, Fax: (32) 2 663 06 40
Sede central en España: Calle Doctor Trueta 113-115, 08005 Barcelona, España, Tel: (34) 93-295-50-00, Fax: (34) 93-295-57-01

1756-6.4.1ES - Octubre de 1999
Reemplaza la publicación 1756-6.4.1 - March de 1999

PN 957259-65
© 1999 Rockwell International Corporation.